

fintech **OS**

Automation Blocks 22.1.1000

User Guide

TOC

Overview	14
Installing SysPacks	15
Prerequisites	16
Pre-Installation Checklist	17
Automatic Installation Steps	17
Post-Installation Setup	19
Manual Import Installation Steps	21
Processor Settings and Mapping	23
OCR and Identity Validation	52
Computer Vision	52
Applications	52
Computer Vision Business Process Model	53
Supported Input Documents	53
Data Security	54
Data Flow	55
Location	56
Compliance	56
Installing Computer Vision	57
1 Install the SysPacks	57
2 Set up the Cognitive Processor subscription key(s)	57
Setting Up a Computer Vision Automation Processor ..	58
1 Create a generic processor settings group	58
2 Add the Computer Vision automation processor to a generic processor settings group	58
3 Configure the automation processor's settings	59

Call the RotatelImage option	59
Computer Vision Settings	60
Examples	68
Computer Vision Mappings	73
Examples	73
Adding Computer Vision to a Digital Journey	74
Examples	76
OCR Processor Field Names	79
Onfido	84
Benefits	85
Applications	85
Installing Onfido	85
Prerequisites	85
1 Install the SysPacks	86
2 Set up the Onfido service subscription key	86
Setting Up an Onfido Automation Processor	87
1 Create a digital flow settings group	87
2 Add the Onfido processor settings to a digital flow settings group	88
Onfido Settings	89
Example	93
Onfido Mappings	95
Example	96
Onfido UI Customization	97
Onfido Request Responses	100
Example	100
Getting Status Changes Notifications Using Webhooks	108
Triggering Identity Verification in a Digital Journey	108
AriadNext	109
Key Features	110

Applications	111
Installing AriadNext	111
1 Install the SysPacks	111
2 Set up the AriadNext service subscription key	112
Setting up the AriadNext Automation Processor	113
1 Create a generic processor settings group	113
2 Add the AriadNext automation processor to a generic processor settings group	114
AriadNext Settings	115
Example	118
AriadNext Request Responses	119
Requests	119
Requests Examples	119
Requests - ADR	158
Requests Examples - ADR	158
AriadNext Workflow Statuses	159
Adding AriadNext to a Digital Journey	160
Face Recognition and Video	162
Face Recognition	162
Face Recognition Processor Features	164
Applications	164
Data Security	164
Data Flow	165
Location	166
Compliance	167
Installing Face Recognition	167
1 Install Server Configuration	167
2 Install Application Configuration	169
Client script on edit mode form	170
Client script on insert mode form	171

3 Upgrade Application Configuration	175
Setting Up a Face Recognition Automation Processor	175
1 Create a digital flow processor settings	175
2 Edit the Face Recognition automation processor	176
Face Recognition Settings	177
Examples	179
Face Recognition Mappings	180
Examples	180
Liveness	181
How the process works	181
How to call constructor Liveness Component	181
How to call the Liveness processor within the Face Recognition processor	182
How to change the text and colour	182
Adding Face Recognition to a Digital Journey	197
Video Streaming	200
Video Streaming Processor Features	201
Applications	201
Installing Video Streaming	202
1 Install the SysPacks	202
2 Set up the Video Streaming Processor subscription key(s)	202
Setting Up a Video Streaming Automation Processor	203
1 Add queues and operators	203
2 Create a digital flow processing settings group	204
3 Add the Video Streaming automation processor to flow settings group	205
4 Configure the automation processor's settings	205
Video Streaming Settings	205
Examples	208
Adding Video Streaming to a Digital Journey	208

Co-browsing	210
Features	211
Security	211
nclInstalling Co-browsing	212
1 Install the SysPacks	212
2 Set up the Co-browsing Service Subscription Key	213
3 Set up the Processor Settings	214
Co-browsing Streaming Flow	215
Adding Co-browsing to a Digital Journey	216
Troubleshooting	218
Document Signing	220
eSign Processor	220
eSign Automation Processor Features	222
Applications	222
Installing eSign	223
Install Server Configuration	223
Install Application Configuration	223
Log in to Innovation Studio:	223
1 Import Packs	223
2 Modify data	223
Log in to Innovation Studio:	224
3 ESign Configuration	224
Processor Settings fields	224
ESign Load component examples	228
4 ESign Download Configuration	228
Processor Settings fields	229
ESign Load component examples	230
5 ESign Configuration for Automatic Signature Profile	230
Auto Profile Case 1	231

Auto Profile Case 2	231
Auto Profile Case 3	232
Upgrade Application Configuration	232
1 Import Again Packs	232
2 Modify data	233
Setting Up an eSign Automation Processor	233
Step 1. Create a digital flow processing settings group	233
Step 2. Add the eSign automation processor to a generic processor settings group	234
Step 3. Configure the automation processor's settings	234
eSign Settings	235
Examples	237
eSign Mappings	238
Examples	239
E-sign with tags or coordinates or both	239
Using workflow library FTOSServices	239
Click2Sign	242
RemoteSign	243
Adding eSign to a Digital Journey	244
Getting Status Changes Notifications Using Webhooks	247
Step 1. Configure webhooks	247
Step 2. Create an endpoint for the webhook	249
Download Envelope Log	251
How to download the envelope log	251
Example	252
Digital Documents Processor	253
Installation	254
Applications	254
Setting Up a Digital Document	254

Creating Document Templates	256
Use token fields	256
Use table tokens	257
Format tables in DOCX and XLSX templates	257
Show or hide objects in document templates	260
Creating Digital Documents	261
Prerequisites	262
Add a digital document using SQL Procedure	262
Add a Digital Document:	265
Define Fetches	266
Set the fetch collection execution order	268
Attaching a Report to the Entity pointing to the Document	269
Using the Document in the Form Driven Flow	272
Automatically Generate Customer Contracts	274
1 Prepare the contract template.	274
2 Set up a digital document based on the contract template.	274
3 Attach a report to the entity based on the digital document.	274
4 Create a button to generate the customer contract.	275
5 How to generate a customer contract from the user interface	276
Campaign Management	277
Omnichannel Campaigns	277
Omnichannel Campaigns Features	278
Applications	278
Installing Omnichannel Campaigns	279
Dependencies	279
Pre-Installation Checklist	279
Installation Steps	280
Creating the Organizational Structure	280
Adding System User Information	281

Adding Business Unit Branches	283
Omnichannel Campaigns Management	284
Managing Campaign Types	284
Adding Campaign Types and Subtypes	285
Editing Campaign Types	287
Deleting Campaign Types	288
Managing Status Reasons	288
Defining Status Reasons	288
Used for Cancelled	291
Used for In Progress	291
Campaign Activity	291
Defining Status Reasons Template	292
Used for Cancelled Status Reason Template	293
Used for In Progress Status Reason Template	293
Campaign Stages	294
Managing Seasonal Campaigns	294
Add seasons	295
Edit seasons	296
Delete seasons	296
Managing Marketing Team Members	296
Add marketing team members	296
Edit marketing team member details	297
Delete marketing team member	298
Creating Campaigns	298
Setting up a Campaign	298
Defining the Campaign Content	301
Marketing Anti-spam Settings	303
Defining the Campaign Audience	304
Scheduling the Campaign in Stages	308
Setup	309

Schedule	311
Campaign Stage Instances (Actual Run)	315
Stage Instance	315
Campaign Activities	317
Execution Errors	318
Distribution	318
Define A/B variations of a stage	323
Refining Audience of a stage	325
Activities of a stage	325
Actions	325
Activities	326
Campaign Activities	326
A/B Control Group of a stage	326
Execution Log of a stage	326
Previewing the Execution Plan	327
Preview campaign stage instances	327
Controlling Campaign Activities	331
Campaign Team Members	331
Viewing Activities	332
Actions	332
Activities	333
Internal Campaign Activities	334
Control Group	334
Saving the Campaign	334
Internal Campaigns	335
Internal Campaign Activities	338
Overview Tab	341
Plan Follow Up Tab	346
Dashboards	346
Viewing Activities Dashboard Charts	349

Filtering Dashboard Data	350
Actions Buttons	351
Creating Multi-Stage Execution Plans	354
Stages Available	359
Execution Tree	359
Multi-Stage Execution Plan Instances	363
Execution Log	365
Campaign Stage Instances	365
Omnichannel Communication Automation	366
Features	366
Installation	366
Applications	367
Sendinblue Email Provider	367
Models	367
Configuration	367
Get API Key	367
Domain and IP	367
Capabilities	368
Transactional	368
Webhooks	368
Create the Channel Configurations	369
Configure Omnichannel Communication Channel Providers	371
1 Apply for subscription key	371
2 Edit the FTOSEmailGateway channel provider	371
Use the Omnichannel Communication Channels	374
Email provider configurations	374
1 Add Communication Channel	374
2 Add message	375
SMS provider configurations	376

Add Communication Channel	376
Channel Provider Statuses	377
Personalized Content Management	378
Content Settings	379
Add Content Settings	379
Managing Personalized Content	381
View Personalized Content Templates	381
Add Personalized Content Templates	381
1 Define content template	382
2 Manage content template items	382
Extend Personalized Content	385
Task Management	388
Task Management	388
Task Management Features	388
Task Management Flow	389
Creating Queues and Tasks	390
Allocating Tasks to Operators	390
Installing Task Management	393
Prerequisites	393
Importing the User Role	393
Install Task Management	393
Task Management Admin Menu	394
Queues	395
Creating a Queue Type	395
Creating a Queue	396
Managing Queue Items	398
Operators	401
Competence Levels	402
Filters	403

Profiles	404
Task Management Dashboard	405
Dashboard Tabs	406
Queue Item Details	407
Processing a Task	407

Overview

SysPacks are a bundle of applications developed in-house through the **High Productivity Fintech Infrastructure (HPFI)** platform. They are exported from the FintechOS Core platform in the form of assets that contain sql, xml, or doc files, and other metadata.

In addition to the [Innovation Studio](#) embedded automation processors, the FintechOS SysPacks bundle comes with automation blocks that allow you to enhance digital journeys and improve customer experience.

Automation blocks or processors are out-of-the-box functionalities used to customize a solution based on your customer's needs. You can use these components for OCR extraction, video streaming, face recognition, marketing campaigns management, electronic signatures, and so on.

FintechOS offers a variety of automation blocks that you can use to customize and enhance digital journeys. For additional details on how to integrate and use FintechOS Automation Blocks see the following pages:

- [OCR and Identity Validation](#)
- [Face Recognition and Video](#)
- [Document Signing](#)
- [Campaign Management](#)
- [Task Management](#)

Installing SysPacks

The steps below describe how to perform both an automatic installation and a manual import of a FintechOS SysPack.

Depending on the FintechOS platform version that you want to install, make sure the correct SysPack type is applied:

1. For standard FintechOS infrastructure installation use Standard SysPacks.
2. For professional/ enterprise FintechOS infrastructure installation use the following SysPacks:
 - a. Banking environments: Professional Banking SysPacks
 - b. Insurance environments: Professional Insurance SysPacks

NOTE

SysPacks are mutually exclusive. The platform installation requires only one SysPack type.

Starting with V20.2.9, SysPacks can be imported asynchronous. Make sure you use async syntax when importing the packages in an Azure environment.

Below are the components for each FintechOS SysPack. Details about each component are in their .zip packages.

Package	Description
00 PreInstall DFP Common	PreInstall DFP Common
01 FTOS DFP Common	N/A
02 FoundationPreInstall	FoundationPreInstall
02 FTOS Content Templates	FTOS Content Templates
02 FTOS Foundation	FTOS Foundation
02 FTOS Versioning	FTOS Versioning PreReq
06 FTOS Project HyperPersonalization	FTOS Hyperpersonalization Processor Data Model
07 FTOS Project Campaign	FTOS Campaign Management Data Model
FTOS AriadNext Processor	N/A
FTOS OCR Processor	FTOS OCR Processor Scripts

Package	Description
FTOS Onfido Processor	FTOS Onfido Processor Scripts
FTOS Project Cognitive Processor Client	N/A
FTOS Project Cognitive Processor Operator	N/A
FTOS Project Data Governance Consent Management	N/A
FTOS Project Data Governance Sensitive Data	FTOS Data Governance Sensitive Data Data Model
FTOS Project Esign Processor	FTOS Esign Processor Scripts
FTOS Project Integration	FTOS Integration Scripts

For an automatic installation, follow the steps described in the **SysPacks Automatic Installation** section.

IMPORTANT!

The HyperPersonalization and Campaign packages must be installed together as they have dependencies.

Prerequisites

In order to install the SysPacks, you need the latest FintechOS platform version installed, with the database configured. For specific steps, see the [Installation](#) page.

NOTE

When using **FtosSysPackageDeployer** with SQL Server Integrated Authentication make sure:

1. The Windows user running the above command has read/ write rights access to the FTOS database.
2. You run the command without the SQL username/ password parameters.

When using **FtosSysPackageDeployer** with SQL Server Build In Authentication make sure:

1. The login used has read/ write access to the FTOS database.
2. You run the above command with the SQL username/ parameters.

Pre-Installation Checklist

The SysPack has unique constraints on some of the standard entities like: FTOS_DFP_FlowSettings, FTOS_DFP_ProcessorSettings, FTOS_VersionSettings, FTOS_VersionSettingsItem, FTOS_EntityStatusSettings, FTOS_MKT_AudienceSegments, FTOS_MKT_Audience.

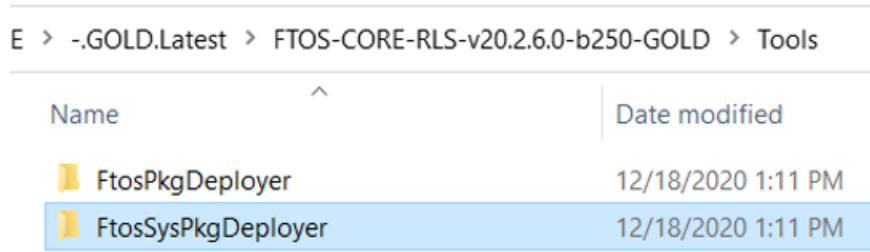
If you have already moved data using the Configuration Data Deployment Package menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

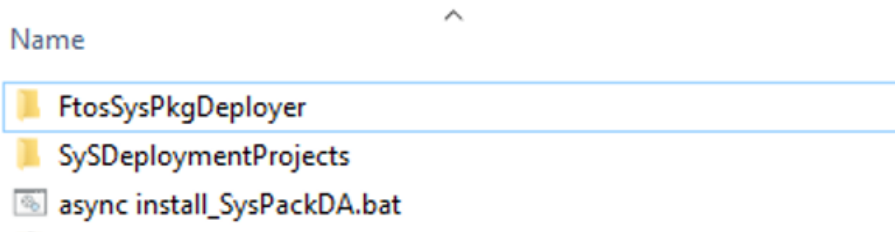
1. Disable the constraints that you have created on your environment, allowing the system to create the new ones after the SysPacks are imported.
2. Use the new **Configuration Data Definitions** imported with the SysPacks when you export the data.

Automatic Installation Steps

1. Download the desired SySDigitalSolutionPackages compatible with your platform version from the [Release Hub](#).
2. Unzip the installation kits.
3. Use *FtosSysPackageDeployer* to install the Syspack as follows:
 - Locate the *FtosSysPackageDeployer* in the unzipped FintechOS installation kit at the following location: `<unzipped_install_archive>\Tools\FtosSysPkgDeployer`.



- Navigate to the location where you have unzipped the SysPack and copy the *FtosSysPackageDeployer* here. Let's call this location **<pckg_deployer_dir>**.



- Open `async install_SysPackDA.bat` to edit and replace the parameters described in the [install_SysPack.bat Parameters Explanation](#) section, with your own values.
- Right-click `async install_SysPackDA.bat` » Run as administrator.

install_SysPackDA.bat Parameters Explanation

For [asynchronous import](#) run the following command:

```
FtosSysPkgDeployer.exe -i -a -s <studio_url> -u <studio_user_name> -p <studio_user_password> -z <db_Server> -v <db_server_login_username> -k <db_server_login_password> -d <db_name> -r <syspack_file_path>
```

Field	Description
<studio_url>	The web URL of the Innovation Studio installation, for example <code>http://localhost/ftos_studio</code> .

Field	Description
<studio_user_name>	The username of the Innovation Studio user under which this import is executed. The user has to exist in Innovation Studio prior to this operation
<studio_user_password>	The password for the Innovation Studio user.
<db_server>	The name of the database server where the FintechOS installation database was created.
<DB_user>	The username of the SQL Server user with administration rights on the FintechOS installation database.
<db_server_login_username>	The login username of the SQL Server user with administration rights on the FintechOS installation database.
<db_server_login_password>	The password for the above mentioned SQL user.
<db_name>	The name of the database where the CoreBanking_3.0.2 is deployed.
<syspack_file_path>	The physical path to the unzipped SysPack previously downloaded.

HINT

For more information about the deployment tool, please run FtosSysPackageDeployer.exe without any arguments to see the built-in help

Post-Installation Setup

After installing the .zip packages, access the 100_AfterImportManualCopy folder and follow the below steps:

1. Add the following images to the Upload EBS folder <portal_EBS_folder> (the Portal with operator flow):
 - a. <syspack_file_path>\100_AfterImportManualCopy
 \CopyToUploadEBS\emptyOCR.jpg
 - b. <syspack_file_path>\100_AfterImportManualCopy\CopyToUploadEBS\emptyPhoto.png

2. Copy the following folders over the FintechOS Portal installation directory for every Portal with back-office or B2C installed.
 - a. <syspack_file_path>\ 100_AfterImportManualCopy \FTOS Project Cognitive Processor Files\dcs-sdk-version\custom
 - b. <syspack_file_path>\ 100_AfterImportManualCopy \FTOS Project Cognitive Processor Files\dcs-sdk-version\custom-on-demand
 - c. Copy any other needed js files in the corresponding js folder.
 - d. For Onfido, follow their instructions from the **InstallGuideOCRWithOnfido v1.1** file.

Cognitive Processor Custom Folders Explanation

Folder	Description
custom	Contains the video custom components: <ul style="list-style-type: none"> • css, images, and javaScripts: dcs-sdk.js and onfido.min.js
custom-on-demand	Contains the liveness component resources.









HINT

For any other information about the steps performed and their result, check `<pkg_deployer_dir>\Logs`.

Manual Import Installation Steps















Follow the below steps if you choose to import the SysPack’s individual deployment packages by hand.

1. Import the projects in SysDigitalSolutionPackages.Log into Innovation Studio and navigate to **Configuration Management > Deployment> Digital Solution Packages**,
2. Click **Import Digital Solution Package** and select the zip packages in the order set by their names and import them one by one.

 07 FTOS Project Campaign.zip	Compressed (zipped) Folder
 06 FTOS Project HyperPersonalization.zip	Compressed (zipped) Folder
 02 FTOS Versioning.zip	Compressed (zipped) Folder
 02 FTOS Foundation.zip	Compressed (zipped) Folder
 02 FTOS Content Templates.zip	Compressed (zipped) Folder
 02 FoundationPreInstall.zip	Compressed (zipped) Folder
 01 FTOS DFP Common.zip	Compressed (zipped) Folder
 00 PreInstall DFP Common.zip	Compressed (zipped) Folder

3. Run the SQL scripts found in the folders that have a part of the name of the packages.

For example **02 FTOS Foundation. zip** and **FTOS Foundation.**

 02 FTOS Versioning.zip	Compressed (zipped) Folder	19 KB	No
 02 FTOS Foundation.zip	Compressed (zipped) Folder	221 KB	No
 02 FTOS Content Templates.zip	Compressed (zipped) Folder	42 KB	No
 02 FoundationPreInstall.zip	Compressed (zipped) Folder	2 KB	No
 01 FTOS DFP Common.zip	Compressed (zipped) Folder	51 KB	No
 00 PreInstall DFP Common.zip	Compressed (zipped) Folder	2 KB	No
 PreInstall DFP Common	File folder		
 FTOS Versioning PreReq	File folder		
 FTOS Onfido Processor Scripts	File folder		
 FTOS OCR Processor Scripts	File folder		
 FTOS Integration Scripts	File folder		
 FTOS Hyperpersonalization Processor Data Mo...	File folder		
 FTOS Foundation	File folder		
 FTOS Esign Processor Scripts	File folder		

NOTE

If you need to update certain packages from the SysPacks, import the .zip files for those packages and run the sql scripts from the folder.

Processor Settings and Mapping

For the below processors, there is a no-code form in the FintechOS Portal where the settings and mappings are defined.

- [Computer Vision](#)
- [eSign](#)
- [Face Recognition](#) with and without Liveness

The settings allow for the following types of controls:

- **String**: creates a normal input control.
- **Number**: creates numeric input control with spinner.
- **Boolean**: creates a drop-down control with following displayed values: Default, Yes, No.
- **Lookup**: creates a control which opens a grid for selecting the desired value. The grid allows filtering.
- **Object**: renders nested controls inside it.
- **Array**: renders nested controls inside it. Add and remove buttons are available to manipulate sets of controls

To render the proper form, follow these steps in the Innovation Studio and in the FintechOS Portal:

1. In the Innovation Studio, open the FTOS_DFP_ProcessorSettings entity .
2. Open the data form, and in the **Advanced** tab fill in the following code:

In the **Before Events** tab:

```
// "settings" in this case represents the name of the entity
attribute
formData.formScope.jsonEditor = new ebs.JsonEditorControl
("settings");
```

In **After Events** tab:

```
var settings = [{}] //will be discussed later

//add listener event on processor type
ebs.addFormChangeEvent("ebsContainerContent",
"digitalProcessorTypeId", setSettingsAndMappingsControls);

async function setSettingsAndMappingsControls(settingsJSON){
  //get the value of processorType which is a GUID
  var digitalProcessorType = ebs.getFormAttributeValue
("ebsContainerContent", "digitalProcessorTypeId");
  //get the optionSetName using the processorTypeId
  var optionSetItemName = await
ebs.getOptionSetItemNameById("FTOS_DFP_
DigitalProcessorTypeOptionSet", digitalProcessorType);
  if(digitalProcessorType) {
    //filter the settings to take the configuration for
the desired processor type
    processorSettingsConfiguration = settings.filter
(function(item){
      return item.key.toLocaleLowerCase() ==
optionSetItemName.toLocaleLowerCase();
    })[0];
    //call the configController method of the editor
created in 'Before events'
    formData.formScope.jsonEditor.configController
(processorSettingsConfiguration.value, settingsJSON);
  }
}

//call the above method with the previously saved JSON in DB
setSettingsAndMappingsControls(ebs.getFormAttributeValue
("ebsContainerContent", "settings"));
```

The settings available are the following :

NOTE

This is an example, please modify as needed.

Settings Example

```
var settings = [
{
  key: 'FaceRecognition',
  value: {
    documentationUrl:
'https://docs.fintechos.com/APs/FaceRecognition/2.0/USERGuide/Content/Settings.htm',
    props: [
      {
        name: "isLiveness",
        label: "Is Liveness",
        type: "boolean"
      },
      {
        name: "DestinationEntityName",
        label: "Destination Entity Name",
        type: "lookup",
        lookupEntityName: "entity",
        lookupViewName: "default",
        displayProp: "displayName"
      },
      {
        name: "SourceEntityName",
        label: "Source Entity Name",
        type: "lookup",
        lookupEntityName: "entity",
        lookupViewName: "default",
        displayProp: "displayName"
      },
      {
        name:
"SourceLookupDestinationName",
        label: "Source Lookup Destination
Name",
        type: "lookup",
        lookupEntityName: "attribute",
```

```

        lookupViewName: "default",
        parentPropertyName:
"SourceEntityName",
        attributeToFilterReference:
"entityid",
        displayProp: "displayName"
    },
    {
        name: "FileAttributeName",
        label: "File Attribute Name",
        type: "string"
    },
    {
        name: "MaxRetry",
        label: "Max Retry",
        type: "number"
    },
    {
        name:
"MinimumAcceptedConfidence",
        label: "Minimum Accepted
Confidence",
        type: "number"
    },
    {
        name: "maskNextStepURLSuccess",
        label: "Mask Next Step URL
Success",
        type: "object",
        items: [
            {
                name: "entity",
                label: "Entity",
                type: "lookup",
                lookupEntityName:
"entity",
                lookupViewName:
"default",
                displayProp:
"displayName"
            },
            {
                name: "form",
                label: "Form",

```

```

        type: "lookup",
        lookupEntityName:
        parentPropertyName:
        attributeToFilterRefe
        lookupViewName:
        displayProp:
    },
    {
        name: "section",
        label: "Section",
        type: "lookup",
        lookupEntityName:
        parentPropertyName:
        attributeToFilterRefe
        lookupViewName:
        displayProp:
    }
]
},
{
    name: "maskNextStepURLFail",
    label: "Mask Next Step URL Fail",
    type: "object",
    items: [
        {
            name: "entity",
            label: "Entity",
            type: "lookup",
            lookupEntityName:
            lookupViewName:
            displayProp:
        }
    ]
}

```

```

        },
        {
            name: "form",
            label: "Form",
            type: "lookup",
            lookupEntityName:
                "entityform",
            parentPropertyName:
                "entity",
            attributeToFilterReference: "entityid",
            lookupViewName:
                "default",
            displayProp:
                "displayName"
        },
        {
            name: "section",
            label: "Section",
            type: "lookup",
            lookupEntityName:
                "entityformsection",
            parentPropertyName:
                "form",
            attributeToFilterReference: "entityFormId",
            lookupViewName:
                "default",
            displayProp:
                "displayName"
        }
    ]
},
{
    name: "businessStatusSuccess",
    label: "Business Status Success",
    type: "string"
},
{
    name: "businessStatusFail",
    label: "Business Status Fail",
    type: "string"
}
}
    ]
}

```

```

    },
    {
      key: 'VideoStreaming',
      value: {
        documentationUrl:
'https://docs.fintechos.com/APs/VideoStreaming/2.0/UserGuide/Content/Settings.htm',
        props: [
          {
            name: "DestinationEntityName",
            label: "Destination Entity Name",
            type: "lookup",
            lookupEntityName: "entity",
            lookupViewName: "default",
            displayProp: "displayName"
          },
          {
            name: "SourceEntityName",
            label: "Source Entity Name",
            type: "lookup",
            lookupEntityName: "entity",
            lookupViewName: "default",
            displayProp: "displayName"
          },
          {
            name:
"SourceLookupDestinationName",
            label: "Source Lookup Destination
Name",
            type: "lookup",
            lookupEntityName: "attribute",
            lookupViewName: "default",
            parentPropertyName:
"SourceEntityName",
            attributeToFilterReference:
"entityid",
            displayProp: "displayName"
          },
          {
            name: "QueueParameters",
            label: "Queue Parameters",
            type: "array",
            items: [
              {

```

```

        name: "ParamName",
        label: "Parameter

Name",
        type: "string",
    },
    {
        name: "ParamValue",
        label: "Parameter

Value",
        type: "string",
    }
]
},
{
    name: "maskNextStepURLSuccess",
    label: "Mask Next Step URL

Success",
    type: "object",
    items: [
        {
            name: "entity",
            label: "Entity",
            type: "lookup",
            lookupEntityName:

"entity",
            lookupViewName:

"default",
            displayProp:

"displayName"
        },
        {
            name: "form",
            label: "Form",
            type: "lookup",
            lookupEntityName:

"entityform",
            parentPropertyName:

"entity",
            attributeToFilterRefe

rence: "entityid",
            lookupViewName:

"default",
            displayProp:

"displayName"
        }
    ]
}
}

```

```

    },
    {
      name: "section",
      label: "Section",
      type: "lookup",
      lookupEntityName:
"entityformsection",
      parentPropertyName:
"form",
      attributeToFilterRefere: "entityFormId",
      lookupViewName:
"default",
      displayProp:
"displayName"
    }
  ],
},
{
  name: "maskNextStepURLFail",
  label: "Mask Next Step URL Fail",
  type: "object",
  items: [
    {
      name: "entity",
      label: "Entity",
      type: "lookup",
      lookupEntityName:
"entity",
      lookupViewName:
"default",
      displayProp:
"displayName"
    },
    {
      name: "form",
      label: "Form",
      type: "lookup",
      lookupEntityName:
"entityform",
      parentPropertyName:
"entity",
      attributeToFilterRefere: "entityid",
    }
  ]
}

```

```

        "default",
        "displayName"
    ],
    "entityformsection",
    "form",
    "reference": "entityFormId",
    "default",
    "displayName"
  ],
  ],
},
{
  key: 'OCR',
  value: {
    documentationUrl:
      'https://docs.fintechos.com/APs/ComputerVision/1.5/UserGuide/Content/Settings.htm',
    props: [
      {
        name: "SourceEntityName",
        label: "Source Entity Name",
        type: "lookup",
        lookupEntityName: "entity",
        lookupViewName: "default",
        displayProp: "displayName"
      },
      {
        name: "Entities",
        label: "Entities",
        lookupViewName:
        displayProp:
      }
    ]
  }
}

```



```

        type: "array",
        "items":[
            {
                name:
                "DestinationEntityName",
                label: "Destination Entity
                Name",
                type: "lookup",
                lookupEntityName:
                "entity",
                lookupViewName: "default",
                displayProp: "displayName"
            },
            {
                name: "SourceEntityName",
                label: "Source Entity
                Name",
                type: "lookup",
                lookupEntityName:
                "entity",
                lookupViewName: "default",
                displayProp: "displayName"
            },
            {
                name:
                "SourceLookupDestinationName",
                label: "Source Lookup
                Destination Name",
                type: "lookup",
                lookupEntityName:
                "attribute",
                parentPropertyName:
                "SourceEntityName",
                attributeToFilterReferenc
                e: "entityid",
                lookupViewName: "default",
                displayProp: "displayName"
            }
        ]
    },
    {
        name: "FileAttributeName",
        label: "File Attribute Name",
        type: "string"
    },
},

```

```

        {
            name: "MaxRetry",
            label: "Max Retry",
            type: "number"
        },
        {
            name: "OptionSets",
            label: "Option Sets",
            type: "array",
            "items": [
                {
                    name:
                    label: "Option Set
                    type: "lookup",
                    lookupEntityName:
                    lookupViewName:
                    displayProp:
                },
                {
                    name:
                    label: "Mapping
                    type: "string"
                },
                {
                    name:
                    label: "Option Set
                    type: "object",
                    items: [
                        {
                            name:
                            label:
                            type:
                    }
                ]
            }
        ]
    }

```

```

EntityName: "optionsetitem",
ViewName: "default",
PropertyName: "OptionSetName",
uteToFilterReference: "optionSetId",
yProp: "displayName"
},
{
  name:
  label:
  type:
  lookup
EntityName: "optionsetitem",
ViewName: "default",
PropertyName: "OptionSetName",
uteToFilterReference: "optionSetId",
yProp: "displayName"
}
]
}
],
{
  name: "LookupEntities",
  label: "Lookup Entities",
  type: "array",
  "items": [
    {
      name:
      label: "Mapping
Name",
  lookup
  lookup
  parent
  attrib
  displa
  lookup
  lookup
  parent
  attrib
  displa
  }
  ]
}
},
{
  name: "LookupEntities",
  label: "Lookup Entities",
  type: "array",
  "items": [
    {
      name:
      label: "Mapping
Name",

```

```

"EntityName",
Name",
"entity",
"default",
"displayName"
type: "string"
},
{
name:
label: "Entity
type: "lookup",
lookupEntityName:
lookupViewName:
displayProp:
},
{
name:
label: "Attribute
type: "lookup",
lookupEntityName:
parentPropertyNam
attributeToFilterR
lookupViewName:
displayProp:
},
{
name: "Parent",
label: "Parent",
type: "object",
items: [
{
name:
label:
type:
"AttributeParentKey",
"Attribute Parent Key",
"lookup",

```

```

EntityName: "attribute",
ViewName: "default",
PropertyName: "EntityName",
uteToFilterReference: "entityid",
yProp: "displayName"
},
{
  name:
  label:
  type:
}
]
},
{
  name: "Validations",
  label: "Validations",
  type: "array",
  "items":[
    {
      name: "type",
      label: "Type",
      type: "string"
    },
    {
      name:
      label:
      type: "string"
    },
    {
      name:

```

```

        label: "Check
Script Name",
        type: "string"
    }
    ]
},
{
    name: "AvailableDocumentTypes",
    label: "Available Document
Types",
    type: "array",
    "items": [
        {
            name: "type",
            label: "Type",
            type: "string"
        },
        {
            name:
"DocumentType",
            label: "Document
Type",
            type: "string"
        },
        {
            name: "Country",
            label: "Country",
            type: "string"
        },
        {
            name: "Provider",
            label: "Provider",
            type: "string"
        }
    ]
},
{
    name:
"maskNextStepURLSuccess",
    label: "Mask Next Step URL
Success",
    type: "object",
    items: [
        {

```

```

"entity",
"default",
"displayName"

"entityform",
e: "entity",
Reference: "entityid",
"default",
"displayName"

"entityformsection",
e: "form",
Reference: "entityFormId",
"default",
"displayName"

    ],
    {
      name: "entity",
      label: "Entity",
      type: "lookup",
      lookupEntityName:
      lookupViewName:
      displayProp:
    },
    {
      name: "form",
      label: "Form",
      type: "lookup",
      lookupEntityName:
      parentPropertyNam
      attributeToFilter
      lookupViewName:
      displayProp:
    },
    {
      name: "section",
      label: "Section",
      type: "lookup",
      lookupEntityName:
      parentPropertyNam
      attributeToFilter
      lookupViewName:
      displayProp:
    }
  }
},
{
  name: "maskNextStepURLFail",

```

```

label: "Mask Next Step URL
Fail",
type: "object",
items: [
  {
    name: "entity",
    label: "Entity",
    type: "lookup",
    lookupEntityName:
    lookupViewName:
    displayProp:
  },
  {
    name: "form",
    label: "Form",
    type: "lookup",
    lookupEntityName:
    parentPropertyName:
    attributeToFilter:
    lookupViewName:
    displayProp:
  },
  {
    name: "section",
    label: "Section",
    type: "lookup",
    lookupEntityName:
    parentPropertyName:
    attributeToFilter:
    lookupViewName:
    displayProp:
  }
]
"entity",
"default",
"displayName"
"entityform",
e: "entity",
Reference: "entityid",
"default",
"displayName"
"entityformsection",
e: "form",
Reference: "entityFormId",
"default",
"displayName"

```



```

        ]
      },
      {
        name:
"businessStatusSuccess",
        label: "Business Status
Success",
        type: "string"
      },
      {
        name: "businessStatusFail",
        label: "Business Status
Fail",
        type: "string"
      },
      {
        name: "DocumentType",
        label: "Document Type",
        type: "string"
      }
    ]
  },
  {
    key: 'ESign',
    value: {
      documentationUrl:
'https://docs.fintechos.com/APs/eSign/2.1/UserGuide/C
ontent/Settings.htm',
      props: [
        {
          name: "EntityName",
          label: "Entity Name",
          type: "lookup",
          lookupEntityName: "entity",
          lookupViewName: "default",
          displayProp: "displayName"
        },
        {
          name: "FileAttributeNameList",
          label: "File Attribute Name List",

```

```

        type: "array",
        items: [
            {
                name:
"fileAttributeName",
                label: "File Attribute
Name",
                type: "string"
            },
            {
                name:
"fileToBeSignedName",
                label: "File To Be Signed
Name",
                type: "string"
            }
        ]
    },
    {
        name: "MaxRetry",
        label: "Max Retry",
        type: "number"
    },
    {
        name: "signedDocumentName",
        label: "Signed Document Name",
        type: "string"
    },
    {
        name: "WebhookUrl",
        label: "Webhook Url",
        type: "string"
    },
    {
        name: "WebhookStatusUrl",
        label: "Webhook Status Url",
        type: "string"
    },
    {
        name: "SignatureSteps",
        label: "Signature Steps",
        type: "array",
        items: [

```

```

        {
            name: "order",
            label: "Order",
            type: "number"
        },
        {
            name: "signatureTag",
            label: "Signature Tag",
            type: "string"
        },
        {
            name:
"signatureTypeTemplate",
            label: "Signature Type
Template",
            type: "string"
        },
        {
            name: "signatureType",
            label: "Signature Type",
            type: "string"
        },
        {
            name: "SignatureData",
            label: "Signature Data",
            type: "array",
            items: [
                {
                    name:
"SourceEntityName",
                    label:
"Source Entity Name",
                    type:
"lookup",
                    lookupEntityName: "entity",
                    lookupViewName: "default",
                    displayPropertyName: "displayName"
                },
                {
                    name:
"SourceLookupDestinationName",

```

```

"Source Lookup Destination Name",
"lookup",
name: "attribute",
type: "default",
entityName: "SourceEntityName",
filterReference: "entityid",
displayProp: "displayName"
},
{
name: "DefaultFields",
label: "Default Fields",
type: "object",
name: "languageCode",
label: "Language Code",
type: "string"
},
{
name: "documentType",
label: "Document Type",
type: "string"
}
],
{
name: "MappedFields",
label:
type:
lookupEntityN
lookupViewNam
parentPropert
attributeToFi
displayProp:
name:
label:
type:
items:[
{
n
l
t
},
{
n
l
t
}
]
},
{
name:

```

```

"Mapped Fields",
"object",

ame: "languageCode",
abel: "Language Code",
ype: "string"

ame: "documentType",
abel: "Document Type",
ype: "string"

ame: "email",
abel: "Email",
ype: "string"

ame: "phoneMobile",
abel: "Phone Mobile",
ype: "string"

ame: "firstName",
abel: "First Name",
ype: "string"

label:
type:
items: [
  {
    n
    l
    t
  },
  {
    n
    l
    t
  },
  {
    n
    l
    t
  },
  {
    n
    l
    t
  }
]

```

```

    },
    {
      name: "lastName",
      label: "Last Name",
      type: "string"
    },
    {
      name: "documentIssuedBy",
      label: "Document Issued By",
      type: "string"
    },
    {
      name: "socialSecurityNumber",
      label: "Social Security Number",
      type: "string"
    },
    {
      name: "documentExpiryDate",
      label: "Document Expiry Date",
      type: "string"
    },
    {
      name: "documentIssuedOn",
      label: "Document Issued On",
      type: "string"
    },
    {
      name: "documentNumber",

```

```

label: "Document Number",
type: "string"
},
],
},
{
  name:
  label:
  type:
},
{
  name:
  label:
  type:
}
],
},
{
  name: "smsText",
  label: "SMS Text",
  type: "string"
},
{
  name: "clickMsg",
  label: "Click link text",
  type: "string"
},
{
  name: "disableEmail",
  label: "Disable Email",
  type: "boolean"
}
],
},
{
  name: "maskNextStepURLSuccess",

```

```

label: "Mask Next Step URL Success",
type: "object",
items: [
  {
    name: "entity",
    label: "Entity",
    type: "lookup",
    lookupEntityName:
"entity",
    lookupViewName:
"default",
    displayProp:
"displayName"
  },
  {
    name: "form",
    label: "Form",
    type: "lookup",
    lookupEntityName:
"entityform",
    parentPropertyName:
"entity",
    attributeToFilterReferenc
e: "entityid",
    lookupViewName:
"default",
    displayProp:
"displayName"
  },
  {
    name: "section",
    label: "Section",
    type: "lookup",
    lookupEntityName:
"entityformsection",
    parentPropertyName:
"form",
    attributeToFilterReferenc
e: "entityFormId",
    lookupViewName:
"default",
    displayProp:
"displayName"
  }
]

```



```

    ]
  },
  {
    name: "maskNextStepURLFail",
    label: "Mask Next Step URL Fail",
    type: "object",
    items: [
      {
        name: "entity",
        label: "Entity",
        type: "lookup",
        lookupEntityName:
"entity",
        lookupViewName:
"default",
        displayProp:
"displayName"
      },
      {
        name: "form",
        label: "Form",
        type: "lookup",
        lookupEntityName:
"entityform",
        parentPropertyName:
"entity",
        attributeToFilterReferenc
e: "entityid",
        lookupViewName:
"default",
        displayProp:
"displayName"
      },
      {
        name: "section",
        label: "Section",
        type: "lookup",
        lookupEntityName:
"entityformsection",
        parentPropertyName:
"form",
        attributeToFilterReferenc
e: "entityFormId",

```

```

        lookupViewName:
        displayProp:
"default",
"displayName"
    }
  ]
},
{
  name: "businessStatusSuccess",
  label: "Business Status Success",
  type: "string"
},
{
  name: "businessStatusFail",
  label: "Business Status Fail",
  type: "string"
},
{
  name: "redirecttoNamirialLink",
  label: "Redirect To Namirial Link",
  type: "string"
}
]
}
]

```

IMPORTANT!

The fields available have to be configured with care. Each field must be tailored to the specific use-case.

To configure the processor itself, follow these steps:

1. Open the FintechOS Portal, **Digital Flow Settings > Flow Settings > Processor Settings** section.
2. To insert a new processor, click on the **Insert** icon on the right- top corner of the screen. to edit an existing processor, select the desired processor from the list.

3. In the Settings tab, a no-code form will be displayed. What has been configured in the code snippet above will be rendered here in a no-code approach.
4. Fill in the name of the processor. Select for each option set provided the needed information e.g. source entity name, type, item etc.

Based on the values added/ selected in each field, it is possible to generate the JSON file. Click the **Preview** button to see the JSON file.

To update the JSON as done for the previous releases, check the "Use inline editor" bool and the Innovation Studio will generate the JSON:

- on Edit mode, the inline editor will be populated with the previously saved value
- on Add mode, the inline editor value is empty.

To consult the documentation on the settings for an automation processor, click the *See documentation* hyperlink which will redirect the user to the documentation website.

The two arrows on the right side of the screen open the editor in full screen to see the whole list of the possible no-code selection.

When trying to save the configurations on Edit mode, however the JSON file cannot be rendered in the UI of the data form due to it having been created using inline editor or the configuration is not correct, a message is displayed below the top buttons above the source entity name in red.

On **Save** mode:

- if the Use inline editor is checked, then the inline editor value is saved
- If the Use inline editor is not checked, then the JSON generated using the UI is saved.

OCR and Identity Validation

Optical character recognition (OCR) technology recognizes and converts the data from images, scanned documents, or identity documents into machine-readable text. The information extracted can be used in identity verification or Know Your Customer (KYC) processes to essentially prevent identity fraud.

FintechOS offers the below components that can be integrated in your digital journeys to further extend their usability.

Computer Vision

The Computer Vision automation processor allows you to automatically populate entity records in your FintechOS applications with text extracted from document scans or photos.

This facilitates business processes such as digital onboarding. Customers can take a picture of their ID card or upload an existing picture from their device. The identity data is captured, parsed, and validated based on the automation processor's settings. Once the scanning completes, the customer's record is automatically populated with his personal information (mappings defined in the Computer Vision automation processor match the various entries in the ID card to specific entity attributes in the data model).

Applications

Computer Vision can be implemented to simplify paper-driven financial or insurance processes, such as:

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes
- Claims handling
- Mortgage processing

Computer Vision Business Process Model

You can choose Optical Character Recognition (OCR) technology based on the [Microsoft Azure Computer Vision OCR API](#).

The OCR services use proprietary recognition models to detect text content from an image and convert the identified text into a machine-readable character stream.

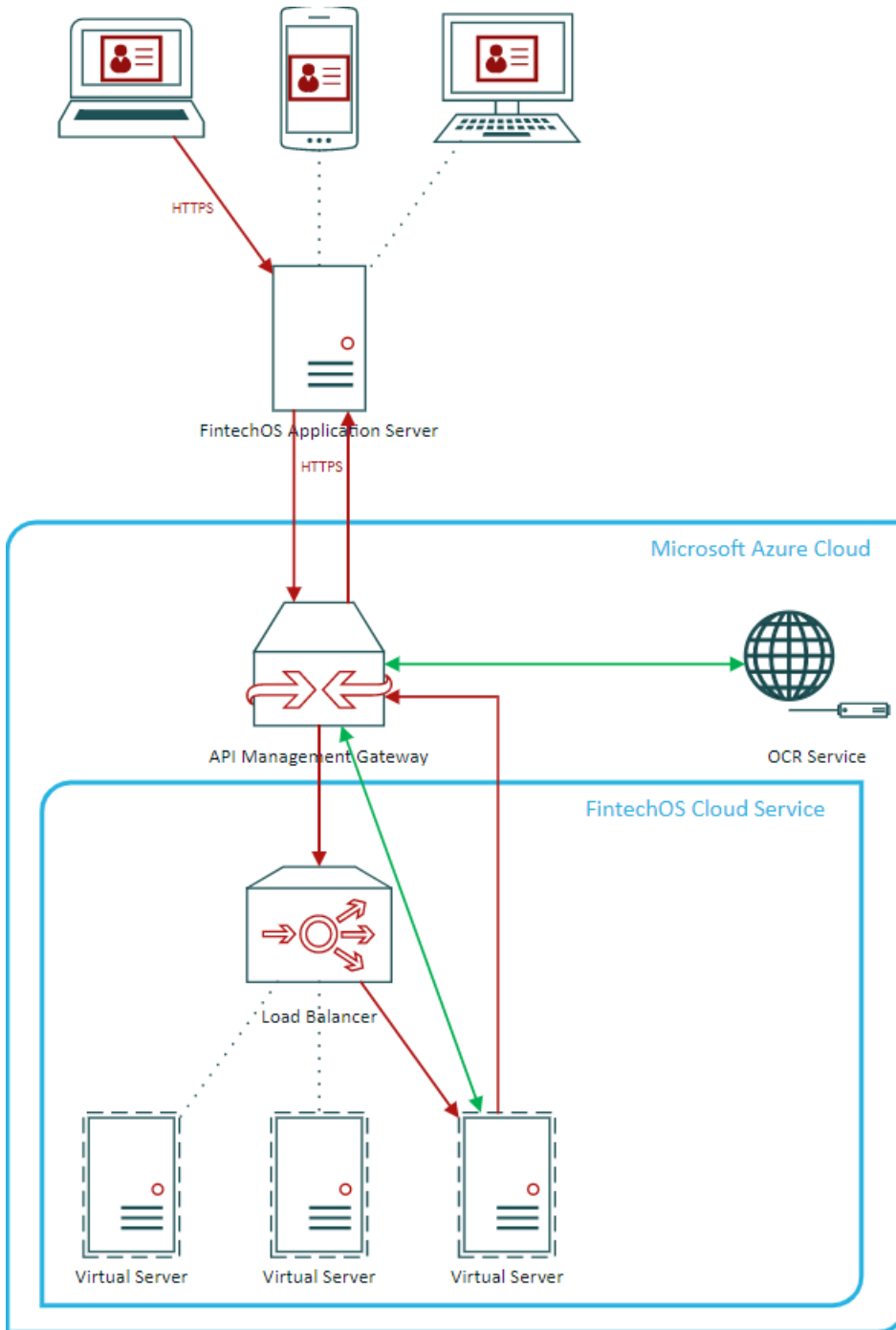
Supported Input Documents

- Documents with a special machine-readable zone (MRZ). For example, Machine Readable Passports usually have an MRZ at the bottom of the identity page at the beginning of the passport.
- Identity cards.
- Driving licenses.

Data Security

All document scans are processed and transferred under strict, GDPR compliant, safety policies.

Data Flow



1. The end-user sends the document scan to the FintechOS application server over secure communication channels (HTTPS encrypted messages, including the HTTP headers and request/response data).
2. The application server sends the document to the FintechOS cloud service via the Azure API Management gateway, also using HTTPS. The API Management gateway ensures secure communication and provides identity and access management to the FintechOS cloud service.
3. The document arrives at the FintechOS cloud service (hosted on a private load balanced cluster of virtual machines in the Azure cloud). The virtual machines are managed by FintechOS and can be accessed only using the API Management services (no Internet access is allowed to any virtual machine or load balanced cluster).
4. The FintechOS cloud service processes the document (each ID card field is delimited) and forwards each field for text recognition to the OCR service (also hosted on the Azure cloud).
5. The OCR service returns the field values to the FintechOS cloud service, which sends the information back to the FintechOS application via the API Management gateway.

No data is stored in cloud. All processed information is immediately deleted.

Location

The API Management gateway, the FintechOS cloud service (load balancer and virtual machines), and the OCR services are provisioned using the Microsoft Azure cloud service in the Western Europe data center (Amsterdam, Netherlands), with fail-over backup services on the Northern Europe data center (Dublin, Ireland). No data leaves the European Union in transit or at rest.

Based on customer requirements, similar services may be provisioned in the future in other regions.

Compliance

For cloud services compliance information, see:

- [Overview of Microsoft Azure compliance](#)
- [Microsoft Services Trust Portal](#)

Installing Computer Vision

1 Install the SysPacks

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.
5. Download the **SySDigitalSolutionPackages v22.1.1000.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page.

2 Set up the Cognitive Processor subscription key(s)

On the FintechOS Portal server, open the *web.config* file in a text editor and add the following entries in the `<appSettings>` section, depending on your subscription keys:

- Generic subscription key for Cognitive Processor services:

```
<add key="FTOSServicesEndpoint" value="URL to the services endpoint"/>
<add key="FTOSServicesAppId" value="service authentication key"/>
```

- Subscription key for Microsoft Azure Computer Vision API:

```
<add key="FTOSServicesOCR2Endpoint" value="URL to the services endpoint"/>
```

```
<add key="FTOSServicesOCR2AppId" value="service authentication key"/>
```

Setting Up a Computer Vision Automation Processor

1 Create a generic processor settings group

The Computer Vision automation processor must be hosted inside a generic processor settings group. A generic processor settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a generic processor settings group you wish to host your Computer Vision automation processor, skip to "[2 Add the Computer Vision automation processor to a generic processor settings group](#)" below. Otherwise, follow the instructions below to create a new generic processor settings group:

1. In FintechOS Portal, click the main menu icon at the top left corner.
2. In the main menu, select **Admin**.
3. Click **Generic Processor Settings**.
4. In the Generic Processor Settings List page, click the Insert button at the top right corner to add a new generic processor settings group.
5. In the Add Generic Processor Settings window, enter a **Name** for your generic processor settings group.
6. Click the **Save and Close** button at the top right corner to save your flow settings group.

2 Add the Computer Vision automation processor to a generic processor settings group

1. In Innovation Studio, click the main menu icon Computer Vision at the top left corner.
2. In the main menu, select **Admin**.

3. Click **Generic Processor Settings**.
4. In the Generic Processor Settings List, double click the generic processor settings group you wish to host your automation processor.
5. In the Edit Generic Processor Setting window, under the Processor Settings region, click the Insert button to add a new automation processor.

3 Configure the automation processor's settings

1. In the **Add Processor Settings** screen, fill in the following fields:
 - Name – Enter a name for your automation processor
 - Digital Processor Type – Select **OCR**.
 - Settings – JSON code for the automation processor's settings. For details, see ["Computer Vision Settings" on the next page](#).
 - Mapping – JSON code for the automation processor's mappings. For details, see ["Computer Vision Mappings" on page 73](#).
2. Click the **Save and Close** button at the top right corner to save your automation processor.

Call the RotateImage option

Create the following request:

```
POST /dcs/ocr2/ocrdocument HTTP/1.1
Host: dcs.fintechos.com
Content-Type: application/json
Ocp-Apim-Subscription-Key: to be obtained from subscription portal
OCRValidations: V01,V10 // the validations separated by ','

{
  "DocumentType": "3", //mandatory - documenttype: MRZ = 1,
  Passport = 2, IdentityCard = 3, DrivingLicence = 4
  "Base64File" : "/9j..", //mandatory - the base64 file
  "RotateImage": true // optional - true/false if you want to
  receive the image rotated based on the text orientation
}
```

Response:

The image will be rotated with an angle that is a multiple of 90° (180°, 270°, etc.), based on the orientation of the text. If the text in the image has an angle smaller than 45°, the image is not be rotated and the Base64File field from the RotatedImage object is null.

```

.....
"PersonalNumber": "111111111",
"GivenName": "asd",
"LastName": "fgh",
"ImageProperties": {
  "Angle": 89.1171 //the angle of the text from the image
},
"RotatedImage": {
  "Base64File": "iVBORw0", //the base64 of the rotated image
  "IsSuccess": true, //true/false if the rotation is ok
  "ErrorMessage": null
},
....

```

Computer Vision Settings

The Computer Vision settings are defined in JSON format as key-value pairs. The following settings are available:

Setting	JSON Key	Description
Workflow entity	SourceEntityName	The entity associated with the business workflow (digital journey) that calls the OCR process. Needed only if the OCR process is used on an edit form (to alter an existing record) to update the workflow entity's business status after the scan (see "BusinessStatusSuccess" on page 67 and "BusinessStatusFail" on page 67). If the OCR process is used on an insert form (to create a new record), this key is not needed.

Setting	JSON Key	Description
Populated entities	Entities	<p>Indicates the entities that will be populated with the scanned values.</p> <ul style="list-style-type: none"> • DestinationEntityName – Name of the entity that is populated with the scanned data. • SourceEntityName – Entity associated with the business workflow (digital journey) that uses the automation processor. <p>If this is the same entity that is populated, this value will be identical to DestinationEntityName.</p> <ul style="list-style-type: none"> • SourceLookupDestinationName – Name of the SourceEntityName lookup key that points to • DestinationEntityName. If they are the same entity, enter the primary key.
User confirmation	WaitUserConfirmation	<p>The default value is true.</p> <ul style="list-style-type: none"> • If true, the user takes or uploads a photo and then clicks Process the Photo to continue the OCR process. • If false, the photo processing is automatically triggered.
Show Upload Photo button	ShowUploadPhotoButton	<ul style="list-style-type: none"> • If true, the Upload Photo button is displayed. • If false, the Upload Photo button is hidden.

Setting	JSON Key	Description
Show Take Picture button	ShowTakePictureButton	<ul style="list-style-type: none"> If true, the Take Picture button is displayed. If false, the camera on your device will not be enabled.
Register face from OCR	RegisterFaceFromOCR	<ul style="list-style-type: none"> If true, the OCR process is successful only if the person's face from their ID is registered. If false, there is not attempt to also register the person's face from their ID.
Session expired minutes	SessionExpiredMins	<p>The time in minutes after which, if no navigation to another screen is done, the session expires. In this case, it redirects the user to a session expired page.</p> <div style="background-color: #e1eef6; padding: 10px; border: 1px solid #d9e1f2;"> <p>NOTE Make sure your main entity's business status is updated to the business status you have set in your flow settings for <code>businessStatusSessionExpired</code>.</p> </div>
Rotate image	RotateImage	<p>Rotates the image to an angle that is multiple of 90° (180, 270 etc.), based on the orientation of the text from the OCR.</p> <p>If the text from the image has an angle smaller than 45°, the image remains the same. If the image doesn't need to be rotated the <code>Base64File</code> field from <code>RotatedImage</code> will be null.</p> <p>The rotated image is stored in the <code>FTOS_DFP_OCR.RotatedImage</code> entity.</p>

Setting	JSON Key	Description
Crop image	CropImage	<p>If true, it extracts the person's face in an image and returns the data in the following structure:</p> <pre>"CroppedImage" : "your_document_field_here", "MatchFound": "your_bit_field_here", "Confidence": "your_text_field_here"</pre>
Maximum number of scan attempts	MaxRetry	<p>The maximum number of scan attempts. If this number of failed scans is reached, the user will be redirected according to the specifications in the "MaskNextStepUrlFail" on page 67.</p>
Option set based validation and replacement	OptionSets	<p>Replaces values returned by the OCR process with entries from an option set.</p> <ul style="list-style-type: none"> • OptionSetName – Name of the option set that stores the valid replacement values. • MappingName – Name of the field as returned by the OCR processor. For details, see "OCR Processor Field Names" on page 79. • OptionSetItems – Key-value pairs that map the value returned by the OCR processor (the key) to the replacement value from the option set (the value). <p>If the value returned by the OCR processor is not found in the OptionSetItems keys, the entry will not be populated. The user will be able to manually select only values from OptionSetName.</p>

Setting	JSON Key	Description
Entity based validation	LookupEntities	<p>Validates values returned by the OCR processor based on records in an entity.</p> <ul style="list-style-type: none"> • MappingName – Name of the field as returned by the OCR processor. For details, see "OCR Processor Field Names" on page 79. • EntityName – Name of the entity that stores the valid values. • AttributeKey – Name of the attribute in EntityName that stores the valid values. <p>Parent – Defines hierarchical relationships between lookup entities. For instance, you can check if a city name belongs to a valid county name.</p> <ul style="list-style-type: none"> • MappingParentName – Name of the parent field as returned by the OCR processor. For details, see "OCR Processor Field Names" on page 79. • AttributeParentKey – Name of the EntityName lookup key linked to the parent entity. <p>If the value returned by the OCR processor is not found in the lookup entity, the entry will not be populated. The user will be able to manually select only values from the lookup entity.</p>

Setting	JSON Key	Description
Advanced validations	Validations	<p>Defines advanced validations based on predefined validation codes.</p> <p>Type – Type of scanned document on which the validation is applied (see "AvailableDocumentTypes" on the next page).</p> <p>Validations – Validation codes. The following validations are available for Romanian identity cards:</p> <ul style="list-style-type: none"> • V01 – Compares the Personal Numeric Code (CNP) control digits extracted from the CNP and MRZ. • V05 – Compares the birth dates extracted from the CNP and MRZ. • V06 – Compares the last 6 CNP digits extracted from the CNP and MRZ. • V07 – Checks if the identity card's expiration date matches the person's birthday. Dates are extracted from the MRZ. • V08 – Checks if the age is above 18 years old. The date of birth is extracted from the MRZ. • V10 – Checks the MRZ integrity per ICAO Document 9303. <p>If the name from the uploaded document image cannot be read from the body or MRZ, an exception is thrown and the following response is returned from DCS:</p> <pre data-bbox="816 1650 1369 1740"> { "IsSuccess": false, </pre>

Setting	JSON Key	Description
		<pre> "ErrorMessage": "ValidatorException - Could not detect name in given document. Please load a valid document", "ExternalErrorMessage": null, "ExternalErrorCode": null } </pre>
Available document types	AvailableDocumentTypes	<p>Defines classifications for the document scans based on the document type and service provider.</p> <ul style="list-style-type: none"> • Type – Assign a name for the class of documents, such as <i>IdRom</i> or <i>passport</i>. • DocumentType – Type of document. Supported values are <i>IdentityCard</i>, <i>MRZ</i>, and <i>DrivingLicence</i>. • Country – Optional parameter for the country code of the document. Currently, supported values are <i>RO</i> and <i>BG</i>. • Provider – OCR provider. Supported value is <i>Azure</i>

Setting	JSON Key	Description
Redirect in case of success	MaskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful scan.</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form.
Redirect in case of failure	MaskNextStepUrlFail	<p>Location in the user interface where the user is redirected after the maximum number of failed scan attempts (see "MaxRetry" on page 63).</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form.
Business status update in case of success	BusinessStatusSuccess	<p>Business workflow status update of the "Workflow entity" on page 60 if the scan is successful.</p> <p>Needed only if the OCR process is used on an edit form (to alter an existing record). If the OCR process is used on an insert form (to create a new record), this key is not needed.</p>
Business status update in case of failure	BusinessStatusFail	<p>Business workflow status update of the "Workflow entity" on page 60 if the scan fails.</p> <p>Needed only if the OCR process is used on an edit form (to alter an existing record). If the OCR process is used on an insert form (to create a new record), this key is not needed.</p>
Document type	DocumentType	<p>Indicates the type of scanning the OCR processor will perform based on the values defined in the "AvailableDocumentTypes" on page 66 setting. Only one document type can be enabled per automation processor.</p>

Examples

Computer Vision settings for an edit form

```

{
  "SourceEntityName": "FTOS_BARET_AccountApplication",
  "Entities": [
    {
      "DestinationEntityName": "FTOS_BNKAP_
RetailApplicantData",
      "SourceEntityName": "FTOS_BARET_
AccountApplication",
      "SourceLookupDestinationName":
"retailApplicantId"
    },
    {
      "DestinationEntityName": "FTOS_BNKAP_
RetailApplicantAddress",
      "SourceEntityName": "FTOS_BARET_
AccountApplication",
      "SourceLookupDestinationName": "FTOS_BNKAP_
RetailApplicantAddressId"
    }
  ],
  "WaitUserConfirmation" : true,
  "ShowUploadPhotoButton" : true,
  "ShowTakePictureButton" : true,
  "RegisterFaceFromOCR" : false,
  "SessionExpiredMins" : 15,
  "RotateImage" : false,
  "CropImage" : false,
  "MaxRetry": 5,
  "OptionSets": [
    {
      "OptionSetName": "FTOS_DFP_GenderOptionSet",
      "MappingName": "Sex",
      "OptionSetItems": {
        "M": "M",
        "F": "F"
      }
    }
  ]
},
"LookupEntities":[
  {

```

```

    "MappingName": "DistrictCode",
    "EntityName": "District",
    "AttributeKey": "Code"
  },
  {
    "MappingName": "BirthDistrictBody",
    "EntityName": "District",
    "AttributeKey": "Code"
  },
  {
    "MappingName": "IssuingCountry",
    "EntityName": "FTOS_CMB_Country",
    "AttributeKey": "code"
  },
  {
    "MappingName": "City",
    "EntityName": "City",
    "AttributeKey": "Name",
    "Parent": {
      "AttributeParentKey": "DistrictId",
      "MappingParentName": "DistrictCode"
    }
  }
],
"Validations": [
  {
    "type": "IdROM",
    "Validations": "V01,V05,V06,V07,V08",
    "CheckScriptName": "ValidateIdROM"
  },
  {
    "type": "Passport",
    "Validations": "V05",
    "CheckScriptName": "ValidateIdPass"
  }
],
"AvailableDocumentTypes": [
  {
    "type": "IdRom",
    "DocumentType": "IdentityCard",
    "Country": "RO",
    "Provider": "Azure"
  },
  {
    "type": "IdBG",

```

```

        "DocumentType": "IdentityCard",
        "Country": "BG",
        "Provider": "Abbyy"
    },
    {
        "type": "Passport",
        "DocumentType": "MRZ",
        "Provider": "Azure"
    },
    {
        "type": "DrivingLicence",
        "DocumentType": "DrivingLicence",
        "Provider": "Azure"
    }
],
"maskNextStepURLSuccess": {
    "entity": "FTOS_BARET_AccountApplication",
    "form": "FTOS_BARET_AccountApplication_UserJourney",
    "section": "Personal Data"
},
"maskNextStepURLFail": {
    "entity": "FTOS_BARET_AccountApplication",
    "form": "FTOS_BARET_AccountApplication_UserJourney",
    "section": "Current Account"
},
"maskNextStepURLFailSessionExpired": {
    "entity": "TestEntity",
    "form": "FTOS_BARET_AccountApplication_UJ_FlowControl",
    "section": "SessionExpired1"
},
"businessStatusSuccess": "OCR Validation",
"businessStatusFail": "OCR Failed",
"DocumentType": "IdRom"
}

```

Computer Vision settings for an insert form

```

{
    "Entities": [
        {
            "DestinationEntityName": "FTOS_ACC_Account",
            "SourceEntityName": "FTOS_ACC_Account",
            "SourceLookupDestinationName": "FTOS_ACC_
Accountid"
        }
    ]
}

```

```

    },
    {
      "DestinationEntityName": "FTOS_ACC_Address",
      "SourceEntityName": "FTOS_ACC_Account",
      "SourceLookupDestinationName":
"DefaultAddressId"
    }
  ],
  "WaitUserConfirmation" : true,
  "ShowUploadPhotoButton" : true,
  "ShowTakePictureButton" : true,
  "RegisterFaceFromOCR" : false,
  "SessionExpiredMins" : 15,
  "RotateImage" : false,
  "CropImage" : false,
  "MaxRetry":5,
  "OptionSets":[
    {
      "OptionSetName": "FTOS_ACC_GenderType",
      "MappingName": "Sex",
      "OptionSetItems": {
        "M": "Male",
        "F": "Female"
      }
    }
  ],
  "LookupEntities": [
    {
      "MappingName":"DistrictCode",
      "EntityName":"District",
      "AttributeKey":"Code"
    },
    {
      "MappingName":"BirthDistrictBody",
      "EntityName":"District",
      "AttributeKey":"Code"
    },
    {
      "MappingName":"IssuingCountry",
      "EntityName":"FTOS_CMB_Country",
      "AttributeKey":"code"
    },
    {
      "MappingName":"City",
      "EntityName":"City",

```

```

        "AttributeKey": "Name",
        "Parent": {
            "AttributeParentKey": "DistrictId",
            "MappingParentName": "DistrictCode"
        }
    },
    ],
    "Validations": [
        {
            "type": "IdRom",
            "Validations": "V01,V05,V06,V07,V08",
            "CheckScriptName": "ValidateIdROM"
        },
        {
            "type": "Passport",
            "Validations": "V05",
            "CheckScriptName": "ValidateIdPass"
        }
    ],
    "AvailableDocumentTypes": [
        {
            "type": "IdRom",
            "DocumentType": "IdentityCard",
            "Country": "RO",
            "Provider": "Azure"
        },
        {
            "type": "IdBG",
            "DocumentType": "IdentityCard",
            "Country": "BG",
            "Provider": "Abbyy"
        },
        {
            "type": "Passport",
            "DocumentType": "MRZ",
            "Provider": "Azure"
        },
        {
            "type": "DrivingLicence",
            "DocumentType": "DrivingLicence",
            "Provider": "Azure"
        }
    ],
    "maskNextStepURLSuccess": {
        "entity": "TestEntity",
    }

```



```

        "form": "FTOS_BARET_AccountApplication_UJ_FlowControl",
        "section": "Personal Data"
    },
    "maskNextStepURLFail": {
        "entity": "TestEntity",
        "form": "FTOS_BARET_AccountApplication_UJ_FlowControl",
        "section": "Current Account"
    },
    "maskNextStepURLFailSessionExpired": {
        "entity": "TestEntity",
        "form": "FTOS_BARET_AccountApplication_UJ_FlowControl",
        "section": "SessionExpired1"
    },
    "DocumentType": "IdRom"
}

```

Computer Vision Mappings

The Automation Blocks mappings match the field names as returned by the OCR processor (keys) with the populated entities' attributes (values).

Setting Name	Description
DocumentsMapping	Holds attribute mappings for various types of scanned documents.
Type	Scanned document type as defined in the list of "Available document types" on page 66 .
Map	Key-value pairs that match the field name as returned by the OCR processor (the key) to the attribute name in the destination entity (the value). See "OCR Processor Field Names" on page 79 and "Populated entities" on page 61 for details.

Examples

Sample JSON code for Automation Blocks mappings

```

{
  "DocumentsMapping": [
    {
      "type": "IdRom",
      "Map": {
        "LastName": "lastName",

```

```

        "GivenName": "firstName",
        "DocumentNumber": "IdCardSeries",
        "StreetType": "streetType",
        "PersonalNumber": "PIN",
        "BirthDate": "dateOfBirth",
        "PlaceOfBirthBody": "placeOfBirth",
        "BirthCountryBody": "birthCountry",
        "Address": "fullAddress",
        "Sex": "gender",
        "DistrictCode": "DistrictId",
        "Nationality": "nationality",
        "City": "CityId",
        "Street": "StreetName",
        "StreetNo": "StreetNo",
        "Storey": "FloorNo",
        "Stairway": "Stairway",
        "ApartmentNo": "ApartmentNo",
        "ApHouse": "BuildingNo",
        "IssuedBy": "IdIssueInstitution",
        "IssuedAt": "IdIssueDate",
        "IssuedUntil": "IdExpirationDate",
        "IssuingCountry": "issuingCountry"
    }
}
]
}

```

Adding Computer Vision to a Digital Journey

1. Open the digital journey in Innovation Studio.
2. Make sure that the form you want to populate includes a button to call the Computer Vision automation processor.
3. Click the **Code** tab.
4. Click the **After Events** subtab.

5. Add the following code in the After Events window:

```
var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = 'FTOS_DFP_OCR';

var params = {};
params.flowSettingsName = '<generic processor settings group name>';
params.processorSettingsType = 'OCR';
params.processorSettingsName = '<processor settings name>';

$('<Computer Vision button name>').click(function () {
    ebs.callActionByName("FTOS_DFP_FlowProcessorSettingsByType", params, function (f) {
        var processorSettingsId =
        f.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
        processorSettingsId, ebs.getCurrentEntityId(), false);
    });
});
```

6. If the Computer Vision automation processor is called in an insert form (to create a new record, not edit an existing record), also add the following code in the After Events window:

```
var ocrResult = sessionStorage.getItem("ocrResult");
ocrResult = JSON.parse(ocrResult);

if (ocrResult) {
    ebs.setFormAttributeValue("ebsContainerContent", "<form field 1 name>", ocrResult.updateObject.<form field 1 name>);
    ebs.setFormAttributeValue("ebsContainerContent", "<form field 2 name>", ocrResult.updateObject.<form field 2 name>);
    .....
    sessionStorage.removeItem("ocrResult");
}
```

This code populates the form fields with the scanned values according to the automation processor's mappings (see ["Computer Vision Mappings" on page 73](#) for details) and clears the scan results from session storage.

7. Click the **Save and Close** button at the top right corner to save your digital journey.

Examples

After Events code for an insert form

In this example:

- We are inserting a record in a generic processor settings group called **Test Flow Setting 1**.
- The name of the Computer Vision automation processor is **OCR_1**.
- The name of the form button that calls the automation processor is **btnOCR1**.
- The Computer Vision automation processor populates the following form fields: **lastName**, **firstName**, and **gender**.

```
var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = 'FTOS_DFP_OCR';

var params = {};
params.flowSettingsName = "Test Flow Setting 1";
params.processorSettingsType = 'OCR';
params.processorSettingsName = 'OCR_1';

$('#btnOCR1').click(function () {
    ebs.callActionByName("FTOS_DFP_FlowProcessorSettingsByType", params, function (f) {
        var processorSettingsId =
            f.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
            processorSettingsId, ebs.getCurrentEntityId(), false);
    });
});

var ocrResult = sessionStorage.getItem("ocrResult");
ocrResult = JSON.parse(ocrResult);

if (ocrResult) {
    ebs.setFormAttributeValue("ebsContainerContent",
        "lastName", ocrResult.updateObject.lastName);
    ebs.setFormAttributeValue("ebsContainerContent",
        "firstName", ocrResult.updateObject.firstName);
}
```

```
ebs.setFormAttributeValue("ebsContainerContent",
"gender", ocrResult.updateObject.gender);
sessionStorage.removeItem("ocrResult");
};
```

After Events code for an edit form

In this example:

- We are editing a record in a generic processor settings group called **Test Flow Setting 2**.
- The name of the Computer Vision automation processor is **OCR_2**.
- The name of the form button that calls the automation processor is **btnOCR2**.
- Since this is an edit form, the Computer Vision automation processor automatically populates the form fields defined in the processor's mapping (for details, see ["Computer Vision Mappings" on page 73](#)).

```
var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = 'FTOS_DFP_OCR';

var params = {};
params.flowSettingsName = "Test Flow Setting 2";
params.processorSettingsType = 'OCR';
params.processorSettingsName = 'OCR_2';

$('#btnOCR2').click(function () {
  ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params, function (f) {
    var processorSettingsId2 =
f.UIResult.Data.ProcessorSettingsId;
    dfpHelper.loadComponent(componentName,
processorSettingsId2, ebs.getCurrentEntityId(), false);
  });
});
```

After Events code for an edit form with two Computer Vision automation processors

In this example:

- We are editing a record in a generic processor settings group called **Test Flow Setting**.
- The form fields are populated in two stages, by two different Computer Vision automation processors called **OCR_1** and **OCR_2**.
- The names of the form buttons that call the automation processors are **btnOCR1** and **btnOCR2**.
- The parameter objects for the automation processors are called **params1** and **params2**.
- Since this is an edit form, the Computer Vision automation processor automatically populates the form fields defined in the processor's mapping (for details, see ["Computer Vision Mappings" on page 73](#)).

```
var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = 'FTOS_DFP_OCR';

var params1 = {};
params1.flowSettingsName = "Test Flow Setting";
params1.processorSettingsType = 'OCR';
params1.processorSettingsName = 'OCR_1';

$('#btnOCR1').click(function () {
    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params1, function (f) {
        var processorSettingsId2 =
f.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
processorSettingsId2, ebs.getCurrentEntityId(), false);
    });
});

var params2 = {};
params2.flowSettingsName = "Test Flow Setting";
params2.processorSettingsType = 'OCR';
params2.processorSettingsName = 'OCR_2';
```

```

$('#btnOCR2').click(function () {
    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params2, function (f) {
        var processorSettingsId2 =
f.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
processorSettingsId2, ebs.getCurrentEntityId(), false);
    });
});

```

OCR Processor Field Names

The OCR processor returns different sets of values, depending on the service provider and type of document. The corresponding field names are described below.

OCR Provider - Azure; Document Type - Identity Card

LastNameBody
 GivenNameBody
 DocumentNumberBody
 StreetType
 StreetName
 PersonalNumberBody
 BirthDateBody
 PlaceOfBirthBody
 BirthCityBody
 BirthDistrictBody
 BirthCountryBody
 Address
 DistrictCode
 City
 Street
 StreetNo
 Storey
 Stairway
 ApartmentNo
 ApHouse
 IssuedBy
 IssuedAt

IssuedUntilBody
IssuingCountry
NationalityBody
Document
Type
Country
DocumentNumber
Nationality
RawBirthDate
BirthDate
Sex
RawIssuedUntil
IssuedUntil
PersonalNumber
GivenName
LastName

OCR Provider - Azure; Document Type - MRZ

Document
Type
Country
DocumentNumber
Nationality
RawBirthDate
BirthDate
Sex
RawIssuedUntil
IssuedUntil
PersonalNumber
GivenName
LastName

OCR Provider - Azure; Document Type - Vehicle Identity Card; Country - RO

IdentityCardNumber
PlateNumber
Type
VehicleTypeDescription
VehicleTypeCategory
Brand

Series
SeriesVariant
SeriesVersion
Model
VehicleIdentificationNumber
OmologationNumber
OwnerName
OwnerSurname
OwnerFullName
OwnerFullAddress
OwnerIsCompany
OwnerCity
OwnerCounty
OwnerStreet
OwnerStreetType
OwnerStreetName
OwnerStreetNo
OwnerStorey
OwnerStairway
OwnerApartmentNo
OwnerApHouse
CurrentUserName
CurrentUserSurname
CurrentUserFullName
CurrentUserFullAddress
CurrentUserIsCompany
CurrentUserCity
CurrentUserCounty
CurrentUserStreet
CurrentUserStreetType
CurrentUserStreetName
CurrentUserStreetNo
CurrentUserStorey
CurrentUserStairway
CurrentUserApartmentNo
CurrentUserApHouse
FirstRegistrationDate
RegisteredCounty
MaxAllowedMass
VehicleWeight
ValidUntil
RegistrationDate
EngineCapacity
Power

FuelType
PowerMassRatio
Color
NumberOfSeats
NumberOfStandingSeats
IdentityCardSeries
IssuedBy
IssuedAt

OCR Provider - Azure; Document Type - Driving Licence

LastName
GivenName
BirthDate
PlaceOfBirth
IssuedAt
IssuedUntil
IssuedBy
PIN
DrivingLicenceNo
DrivingCategory
Address

OCR Provider - Abbyy; Document Type - Identity Card; Country - RO

LastNameBody
GivenNameBody
DocumentNumberBody
StreetType
PersonalNumberBody
BirthDateBody
PlaceOfBirthBody
BirthCountryBody
Validations
ValidationsString
Address
BirthDate
GivenName
LastName

Sex
PersonalNumber
DocumentNumber
DistrictCode
Nationality
City
Street
StreetNo
Storey
Stairway
ApartmentNo
ApHouse
IssuedBy
IssuedAt
IssuedUntil
IssuingCountry

OCR Provider - Abbyy; Document Type - Identity Card; Country - BG

FatherName
LastNameLatin
GivenNameLatin
FatherNameLatin
District
Municipality
PlaceOfBirth
HeightAndColorOfEyes
Height
ColorOfEyes
Optional1
Optional2
Document
Type
Country
DocumentNumber
Nationality
RawBirthDate
BirthDate
Sex
RawIssuedUntil
IssuedUntil

PersonalNumber
GivenName
LastName

OCR Provider - Abbyy; Document Type - MRZ

LastNameBody
GivenNameBody
DocumentNumberBody
StreetType
PersonalNumberBody
BirthDateBody
PlaceOfBirthBody
BirthCountryBody

Onfido

Onfido helps you validate a user's identity by checking the authenticity of the submitted identity document. The solution scans the ID front and back to validate the name introduced by the user with data extracted from the document. If, for example, there's a middle name on the document that's not specified by the user, then the data doesn't match and the check can fail. In addition to the data extracted from the document scans Onfido retrieves the breakdowns and validations that depend on each configuration. This information is stored in the FTOS_ONFIDO_CHECKREPORT defined in JSON format.

Apart from identity documents validation, the Onfido automation processor offers face similarity checks by comparing a clear photo or a video provided by the applicant to their identity document. Using the face similarity functionality, it quickly verifies that the applicant is a real person and they are the same as the person from the identity document.

The data that Onfido handles must be GDPR compliant. To delete data from Onfido, you must do so manually from the Onfido account.

Benefits

- **Digitization.** Validate your users' digital accounts based on their real identities.
- **Customer Acquisition.** Streamline your customer onboarding process with a simple user experience.
- **Fraud Detection.** Prevent identity fraud with document verification.
- **Security.** Manual checks for warning cases on top of automated validations.
- **Compliance.** Implement KYC and AML requirements at scale.

Applications

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes
- Claims handling
- Mortgage processing

Installing Onfido

Prerequisites

Before installation, you need:

- An active FintechOS account. For details, contact a FintechOS sales representative.
- Make sure you have the *dcs-sdk-onfido.js* installed on your environment in your custom folder. For details, contact a FintechOS sales representative.

1 Install the SysPacks

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.
5. Download the **SySDigitalSolutionPackages v22.1.1000.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page,

2 Set up the Onfido service subscription key

IMPORTANT!

Make sure that the Vault configurations are done and that the [Webhook](#) is created.

In Vault

Key Path	Key Name	Value
kv/<environment>/<Fintech OS Portal instance>/app-settings	FTOSServiceOnfidoEndpoint	DCS web app endpoint URL provided by FintechOS.
kv/<environment>/<Fintech OS Portal instance>/app-settings	FTOSServicesOnfidoAppId	ID for the Onfido service subscription.

Key Path	Key Name	Value
kv/<environment>/<Fintech OS Portal instance>/app-settings	FTOSServicesOnfidoSubscription Key	Subscription key for the Onfido service.

DCS Onfido configurations are stored using **Vault** secrets inside a KV (key-value) Service, using the following structure path: `kv/dcs/kyc-onfido`.

(Deprecated) Adding Keys In web.config

Key	Value
FTOSServiceOnfidoEndpoint	DCS web app endpoint URL provided by FintechOS.
FTOSServicesOnfidoAppId	ID for the Onfido service subscription.
FTOSServicesOnfidoSubscriptionKey	Subscription key for the Onfido service.

```
<add key="FTOSServicesOnfidoEndpoint" value="https://"/>
<add key="FTOSServicesOnfidoAppId" value=""/>
<add key="FTOSServicesOnfidoSubscriptionKey" value=""/>
```

Setting Up an Onfido Automation Processor

1 Create a digital flow settings group

The Automation Blocks automation processor must be hosted inside a digital flow settings group. A digital flow settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a digital flow settings group you wish to host your Onfido automation processor, skip to "[2 Add the Onfido processor settings to a digital flow settings group](#)" on the next page. Otherwise, follow the instructions below to create a new digital flow settings group:

1. In Innovation Studio, go to **Main Menu > Digital Experience > Digital Journeys > Processor Settings**.
2. In the Digital Flow Settings list page, click the **Insert** button at the top right corner to add a new digital flow settings group.
3. In the Add Digital Flow Settings window, enter a **Name** for your digital flow settings group.
4. If you already have a digital journey set up where you wish to call the Onfido automation processor, select it from the Digital Journey drop-down box.
5. Click the **Save and Close** button at the top right corner to save your flow settings group.

2 Add the Onfido processor settings to a digital flow settings group

1. In Innovation Studio, **Main Menu > Digital Experience > Digital Journeys > Processor Settings**.
2. In the Digital Flow Settings list, double click the desired digital flow settings group.
3. In the Processor Settings section, click the **Insert** button.
4. Fill in the following fields:
 - Name – Enter a name for your processor settings.
 - Flow Settings – Leave the default value.
 - Digital Processor Type – Select **Onfido**.
 - Settings – JSON code for the automation processor's settings. For details, see ["Onfido Settings" on the next page](#).
 - Mapping – JSON code for the automation processor's mappings. For details, see ["Onfido Mappings" on page 95](#).
5. Click **Save and Close** at the top right corner of the screen.

Onfido Settings

The Onfido settings are defined in JSON format as key-value pairs. The following settings are available:

JSON Key	Description
SourceEntityName	The entity associated with the digital journey that calls the Onfido processor. Needed only if the Onfido processor is used on an edit form (to alter an existing record) to update the workflow entity's business status after the scan. If the Onfido processor is used on an insert form (to create a new record), this key is not needed.
UseProcessorForOCR	Boolean value. When set to true, an OCR request is sent to DCS.
UseProcessorForCheck	Boolean value. When set to true, the raw identity check reports received from the Onfido service are attached to the FTOS_ONFIDO_CHECK entity (in the FTOS_ONFIDO_CHECKREPORT related entity).
UploadFallback	Boolean value - default: false. When set to true, an optimized camera UI is used to take a live photo of the identity document. When this is not possible (because of an unsupported browser or mobile device), it falls back to the mobile device's default camera application. <div style="background-color: #f4a460; padding: 10px; border-radius: 5px;"> <p>IMPORTANT! This method does not guarantee live capture, because some mobile device browsers and camera applications allow uploads from the user's gallery of photos.</p> </div>
OnfidoCountry	The identity document's issuing country specified using a 3-letter ISO 3166-1 alpha-3 country code.
OnfidoDocumentType	An array with desired document types. Allowed values are: "national_identity_card", "passport", "driving_licence", and "residence_permit".
Referrer	The URL of the web page where the Web SDK is used. The referrer sent by the browser must match the referrer URL pattern in the SDK token for the SDK to successfully authenticate.
DocumentHasTwoSides	Boolean value indicating if the user must scan both sides of the identity document.

JSON Key	Description
UseDefaultValues	Boolean value. When set to <i>true</i> , the processor creates a new Onfido applicant with the information defined in " DefaultApplicantData " below.
UseFaceSimilarity	Boolean value. When set to <i>true</i> , the Face Similarity Type field is displayed to validate face similarity.
FaceSimilarityType	<p>Displayed if the Use Face Similarity field is set to <i>true</i>. The following options are available</p> <ul style="list-style-type: none"> • Standard, in which you take a simple selfie. • Video, in which you need to make the requested moves for validation.
DefaultApplicantData	JSON object containing key-value pairs for the default Onfido applicant's data.
ApplicantData	The attributes from which the Onfido applicant's data is retrieved.
Entities	<p>Indicates the entities that populates with the scanned values.</p> <ul style="list-style-type: none"> • DestinationEntityName – Name of the entity that is populated with the scanned data. • SourceEntityName – Entity associated with the digital journey that uses the automation processor. If this is the same entity that is populated, this value is identical to DestinationEntityName. • SourceLookupDestinationName – Name of the SourceEntityName lookup key that points to DestinationEntityName. If they are the same entity, enter the primary key.

JSON Key	Description
OptionSets	<p>Replaces values returned by the Onfido processor with entries from an option set.</p> <ul style="list-style-type: none"> • OptionSetName – Name of the option set that stores the valid replacement values. • MappingName – Name of the field as returned by the Onfido processor. • OptionSetItems – Key-value pairs that map the value returned by the Onfido processor (the key) to the replacement value from the option set (the value). <p>If the value returned by the Onfido processor is not found in the OptionSetItems keys, the entry is not be populated. The user is able to manually select only values from OptionSetName.</p>

JSON Key	Description
LookupEntities	<p>Validates values returned by the Onfido processor based on records in an entity.</p> <ul style="list-style-type: none"> • MappingName – Name of the field as returned by the Onfido processor. • EntityName – Name of the entity that stores the valid values. • AttributeKey – Name of the attribute in EntityName that stores the valid values. <p>Parent – Defines hierarchical relationships between lookup entities. For instance, you can check if a city name belongs to a valid county name.</p> <ul style="list-style-type: none"> • MappingParentName – Name of the parent field as returned by the Onfido processor. • AttributeParentKey – Name of the EntityName lookup key linked to the parent entity. <p>If the value returned by the Onfido processor is not found in the lookup entity, the entry does not populate. The user is able to manually select only values from the lookup entity.</p>
MaskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful check.</p> <ul style="list-style-type: none"> • entity – Entity name. • form – Form name of the above entity. • section – Optional parameter for the section name of the above form.

JSON Key	Description
MaskNextStepUrlFail	<p>Location in the user interface where the user is redirected after a failed check.</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form.
UseLocalization	<p>Boolean value. Setting it to <i>true</i> enables user interface localization. Localization uses the FTOS_DFP_OnfidoLocalization client script library to populate the interface text fields. By default, only the en-US locale is defined. Edit the library to add additional languages.</p>
DocumentType	<p>The document type that identifies the set of attribute mappings used by the processor (see "Onfido Mappings" on page 95). Only one document type can be enabled per automation processor.</p>

Example

```

{
  "SourceEntityName": "FTOS_BARET_AccountApplication",
  "UseProcessorForOCR": true,
  "UseProcessorForCheck": true,
  "UploadFallback": true,
  "OnfidoCountry": "ROU",
  "OnfidoDocumentType": ["national_identity_card"],
  "Referrer": "https://myOnfidoPortal.net/",
  "DocumentHasTwoSides": false,
  "UseDefaultValues": true,
  "UseFaceSimilarity": true,
  "FaceSimilarityType": "standard" / "video",
  "DefaultApplicantData": {
    "LastName": "DefaultLastName",
    "FirstName": "DefaultFirstName",
    "Email": "DefaultEmail@email.com"
  },
  "ApplicantData": {
    "LastName": {
      "AttributeName": "retailApplicantId_lastName",
      "Extension": "retailApplicantData"
    }
  },
}

```

```

    "FirstName": {
      "AttributeName": "retailApplicantId_firstName",
      "Extension": "retailApplicantData"
    },
    "Email": {
      "AttributeName": "retailApplicantId_email",
      "Extension": "retailApplicantData"
    }
  },
  "Entities":[
    {
      "DestinationEntityName":"FTOS_BNKAP_RetailApplicantData",
      "SourceEntityName":"FTOS_BARET_AccountApplication",
      "SourceLookupDestinationName":"retailApplicantId"
    },
    {
      "DestinationEntityName":"FTOS_BNKAP_
RetailApplicantAddress",
      "SourceEntityName":"FTOS_BARET_AccountApplication",
      "SourceLookupDestinationName":"FTOS_BNKAP_
RetailApplicantAddressId"
    }
  ],
  "OptionSets":[
    {
      "OptionSetName":"Gender Type",
      "MappingName":"Sex",
      "OptionSetItems":{
        "Male":"Male",
        "Female":"Female"
      }
    }
  ],
  "LookupEntities":[
    {
      "MappingName":"DistrictCode",
      "EntityName":"District",
      "AttributeKey":"Code"
    },
    {
      "MappingName":"BirthDistrictBody",
      "EntityName":"District",
      "AttributeKey":"Code"
    },
    {
      "MappingName":"IssuingCountry",

```

```

        "EntityName": "FTOS_CMB_Country",
        "AttributeKey": "code"
    },
    {
        "MappingName": "City",
        "EntityName": "City",
        "AttributeKey": "Name",
        "Parent": {
            "AttributeParentKey": "DistrictId",
            "MappingParentName": "DistrictCode"
        }
    }
]
,
    "maskNextStepURLSuccess": {
        "entity": "FTOS_BARET_AccountApplication",
        "form": "TestOnfido",
        "section": "step2"
    },
    "maskNextStepURLFail": {
        "entity": "FTOS_BARET_AccountApplication",
        "form": "TestOnfido",
        "section": "Step1"
    },
    "UseLocalization": true,
    "DocumentType": "IdRom"
}

```

Onfido Mappings

The Onfido mappings match the field names as returned by the Onfido processor (keys) with the populated entities' attributes (values).

Setting Name	Description
DocumentsMapping	Holds attribute mappings for various types of scanned documents.
Type	Scanned document type.
Map	Key-value pairs that match the field name as returned by the Onfido processor (the key) to the attribute name in the destination entity (the value).

Example

The *side* key indicates which side of the identity document the field belongs to. For one-sided documents, set it always to *front*.

```
{
  "DocumentsMapping":
  [
    {
      "type": "IdRom",
      "Map":
      {
        "PictureAttribute": { "attribute": "imgFaceOCR",
"side": "front" },
        "LastName": { "attribute": "lastName", "side":
"front" },
        "GivenName": { "attribute": "firstName", "side":
"front" },
        "FullName": { "attribute": "fullName", "side":
"back" },
        "SpouseName": { "attribute": "spouseName", "side":
"back" },
        "WidowName": { "attribute": "widowName", "side":
"back" },
        "AliasName": { "attribute": "aliasName", "side":
"back" },
        "TypeOfDocument": { "attribute": "typeOfDocument",
"side": "back" },
        "DocumentNumber": { "attribute": "IdCardSeries",
"side": "front" },
        "StreetType": { "attribute": "streetType", "side":
"front" },
        "PersonalNumber": { "attribute": "PIN", "side":
"front" },
        "BirthDate": { "attribute": "dateOfBirth", "side":
"front" },
        "PlaceOfBirthBody": { "attribute": "placeOfBirth",
"side": "front" },
        "BirthCountryBody": { "attribute": "birthCountry",
"side": "front" },
        "Address": { "attribute": "fullAddress", "side":
"front" },
        "Sex": { "attribute": "gender", "side": "front" },
        "DistrictCode": { "attribute": "DistrictId",
"side": "front" },
        "Nationality": { "attribute": "nationality",
"side": "front" },
      }
    }
  ]
}
```



```

        "IssuedBy": { "attribute": "IssueInstitution",
"side": "front" },
        "IssuedAt": { "attribute": "IdIssueDate", "side":
"front" },
        "IssuedUntil": { "attribute": "IdExpirationDate",
"side": "front" },
        "IssuingCountry": { "attribute": "issuingCountry",
"side": "front" },
        "Confidence": { "attribute": "confidence", "side":
"front" },
        "mrz_line1": { "attribute": "mrz_line1", "side":
"front" },
        "mrz_line2": { "attribute": "mrz_line2", "side":
"front" },
        "mrz_line3": { "attribute": "mrz_line3", "side":
"front" },
        "AddressLine1": { "attribute": "AddressLine1",
"side": "front" },
        "AddressLine2": { "attribute": "AddressLine2",
"side": "front" },
        "AddressLine3": { "attribute": "AddressLine3",
"side": "front" },
        "AddressLine4": { "attribute": "AddressLine4",
"side": "front" },
        "AddressLine5": { "attribute": "AddressLine5",
"side": "front" }
    }
}
]
}

```

Onfido UI Customization

The Automation Blocks component supports language and UI customization through the language and customUI parameters.

- language: Customize the language displayed on the SDK by passing a string for the supported languages or object. The following languages are supported: en_US, de_DE, es_ES, fr_FR, it_IT, pt_PT, nl_NL.
For unsupported languages, send an object containing local string, a local tag and phrases, and an object containing the keys you want to override. For

example:

```
{
  locale: 'en_US',
  phrases: {
    "welcome": {
      "next_button": "Select document",
      "subtitle": "This will take a few minutes.",
      "title": "Validate identity."
    }
  }
}
```

- customUI: Customize text, the SDK main container, buttons, links, icon background color, and pop-ups by using the following properties:

```
// Typography
fontFamilyTitle: string;
fontFamilySubtitle: string;
fontFamilyBody: string;
fontSizeTitle: string;
fontSizeSubtitle: string;
fontSizeBody: string;
fontWeightTitle: string;
fontWeightSubtitle: string;
fontWeightBody: string;
colorContentTitle: string;
colorContentSubtitle: string;
colorContentBody: string;
// Modal
colorBackgroundSurfaceModal: string;
colorBorderSurfaceModal: string;
borderWidthSurfaceModal: string;
borderStyleSurfaceModal: string;
borderRadiusSurfaceModal: string;
// Buttons
colorContentButtonPrimaryText: string;
colorBackgroundButtonPrimary: string;
colorBackgroundButtonPrimaryHover: string;
colorBackgroundButtonPrimaryActive: string;
colorBorderButtonPrimary: string;
```

```
colorContentButtonSecondaryText: string;
colorBackgroundButtonSecondary: string;
colorBackgroundButtonSecondaryHover: string;
colorBackgroundButtonSecondaryActive: string;
colorBorderButtonSecondary: string;
colorContentDocTypeButton: string;
colorBackgroundDocTypeButton: string;
colorBorderDocTypeButton: string;
colorBorderDocTypeButtonHover: string;
colorBorderDocTypeButtonActive: string;
colorBackgroundIcon: string;
// Shared Buttons
borderRadiusButton: string;
buttonGroupStacked: boolean;
// Links
colorContentLinkTextHover: string;
colorBorderLinkUnderline: string;
colorBackgroundLinkHover: string;
colorBackgroundLinkActive: string;
// Warning Popups
colorContentAlertInfo: string;
colorBackgroundAlertInfo: string;
colorContentAlertInfoLinkHover: string;
colorContentAlertInfoLinkActive: string;
// Error Popups
colorContentAlertError: string;
colorBackgroundAlertError: string;
colorContentAlertErrorLinkHover: string;
colorContentAlertErrorLinkActive: string;
// Info Header/Highlight Pills
colorBackgroundInfoPill: string;
colorContentInfoPill: string;
// Icon Buttons
colorBackgroundButtonIconHover: string;
colorBackgroundButtonIconActive: string;
// Camera Shutter Button
colorBackgroundButtonCameraHover: string;
colorBackgroundButtonCameraActive: string;
```

Onfido Request Responses

The Onfido OCR processor is composed of a client side script, automation scripts, automation script libraries, and a client script library. When the applicant's documents are validated, the call is made using the below method. The OCR data is then saved in the entity based on the [Settings](#) configurations and the response is returned in JSON format.

NOTE

When the Onfido automation processor is called, the response returned is saved in JSON format in the customer's FintechOS Digital Journey. The parsing of the returned JSON object depends on the implementation team.

- `getReport`: the result contains the report for a related check.

Example

Sample JSON code for Onfido `getReport` response

```
{
  "id": "571ddd12-5e9a-43c9-ad87-3c2f434ece9b",
  "created_at": "2022-04-27T14:14:34Z",
  "name": "document",
  "href": "/v3.2/reports/571ddd12-5e9a-43c9-ad87-3c2f434ece9b",
  "status": "complete",
  "result": "consider",
  "sub_result": "suspected",
  "breakdown": {
    "age_validation": {
      "breakdown": {
        "minimum_accepted_age": {
          "properties": {

          },
          "result": "clear"
        }
      }
    },
    "result": "clear"
  },
  "compromised_document": {
```

```
    "result":"clear"
  },
  "data_comparison":{
    "breakdown":{
      "date_of_birth":{
        "properties":{

        },
        "result":null
      },
      "date_of_expiry":{
        "properties":{

        },
        "result":null
      },
      "document_numbers":{
        "properties":{

        },
        "result":null
      },
      "document_type":{
        "properties":{

        },
        "result":null
      },
      "first_name":{
        "properties":{

        },
        "result":null
      },
      "gender":{
        "properties":{

        },
        "result":null
      },
      "issuing_country":{
        "properties":{

        },
        "result":null
      }
    }
  }
}
```

```

    },
    "last_name":{
      "properties":{

      },
      "result":null
    }
  },
  "result":null
},
"data_consistency":{
  "breakdown":{
    "date_of_birth":{
      "properties":{

      },
      "result":"clear"
    },
    "date_of_expiry":{
      "properties":{

      },
      "result":"clear"
    },
    "document_numbers":{
      "properties":{

      },
      "result":"clear"
    },
    "document_type":{
      "properties":{

      },
      "result":"clear"
    },
    "first_name":{
      "properties":{

      },
      "result":"clear"
    },
    "gender":{
      "properties":{

```

```

    },
    "result": "clear"
  },
  "issuing_country": {
    "properties": {
      },
      "result": "clear"
    },
    "last_name": {
      "properties": {
        },
        "result": "clear"
      },
      "multiple_data_sources_present": {
        "properties": {
          },
          "result": null
        },
        "nationality": {
          "properties": {
            },
            "result": null
          }
        },
        "result": "clear"
      },
      "data_validation": {
        "breakdown": {
          "barcode": {
            "properties": {
              },
              "result": null
            },
            "date_of_birth": {
              "properties": {
                },
                "result": "clear"
              },
              "document_expiration": {

```

```

        "properties":{
        },
        "result":"consider"
    },
    "document_numbers":{
        "properties":{
            "document_number":"clear",
            "personal_number":"clear"
        },
        "result":"clear"
    },
    "expiry_date":{
        "properties":{
        },
        "result":"clear"
    },
    "gender":{
        "properties":{
        },
        "result":"clear"
    },
    "mrz":{
        "properties":{
        },
        "result":"consider"
    }
},
"result":"consider"
},
"image_integrity":{
    "breakdown":{
        "colour_picture":{
            "properties":{
            },
            "result":"clear"
        },
        "conclusive_document_quality":{
            "properties":{
                "abnormal_document_features":"clear",
                "corner_removed":"clear",
            }
        }
    }
}

```



```

        "digital_document":"clear",
        "missing_back":"clear",
        "obscured_data_points":"clear",
        "obscured_security_features":"clear",
        "punctured_document":"clear",
        "watermarks_digital_text_overlay":"clear"
    },
    "result":"clear"
},
"image_quality":{
    "properties":{

    },
    "result":"clear"
},
"supported_document":{
    "properties":{

    },
    "result":"clear"
}
},
"result":"clear"
},
"issuing_authority":{
    "breakdown":{
        "nfc_active_authentication":{
            "properties":{

            },
            "result":null
        },
        "nfc_passive_authentication":{
            "properties":{

            },
            "result":null
        }
    },
    "result":null
},
"police_record":{
    "result":null
},
"visual_authenticity":{

```

```
"breakdown":{
  "digital_tampering":{
    "properties":{

    },
    "result":"clear"
  },
  "face_detection":{
    "properties":{

    },
    "result":"clear"
  },
  "fonts":{
    "properties":{

    },
    "result":"consider"
  },
  "original_document_present":{
    "properties":{
      "document_on_printed_paper":"clear",
      "photo_of_screen":"consider",
      "scan":"clear",
      "screenshot":"clear"
    },
    "result":"consider"
  },
  "other":{
    "properties":{

    },
    "result":"consider"
  },
  "picture_face_integrity":{
    "properties":{

    },
    "result":"consider"
  },
  "security_features":{
    "properties":{

    },
    "result":"consider"
  }
}
```


Getting Status Changes Notifications Using Webhooks

Webhooks are automated message sent from an application when an event is triggered. In the Onfido dashboard, webhooks need to be defined allowing you to get notifications when the applicant document check and report are completed.

NOTE

Only one webhook can be created for each client account in the Onfido dashboard.

In the Automation Blocks [settings](#), if the setting key `UseProcessorForCheck` is set to `true`, a new record is created in the `FTOS_OnfidoIntegration` entity. In this entity, the following entities are attached:

- `FTOS_ONFIDO_CHECK`: for document check that contains the **Pending**, **Webhook Received**, and **Completed** statuses.
- `FTOS_ONFIDO_CHECKREPORT`: for report check that stores the **Pending**, **Webhook Received**, and **Completed** statuses. It contains the raw response for each report that is received from Onfido.

When a new check is created, the default status is **submitted**. Then, the `FTOS_Onfido_WebHook` is called from DCS, the report is saved, and the status is changed to **completed**.

Triggering Identity Verification in a Digital Journey

The Onfido SDK is responsible for capturing and uploading photos or videos. The Onfido automation processor can be added to a Digital Journey and further enhanced via Innovation Studio based on the customer's business needs.

Follow the below steps when adding this automation processor to your Digital Journey:

1. In Innovation Studio, open the customer's form driven flow in the editor.
2. Navigate to the form or form step where you wish to initiate the identity check.

3. In the After Events tab, add code based on the model below:

```

var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = 'FTOS_DFP_Onfido'; //name of the
componentvar
var recordId = formData.id;
var flowSettingsName = formData.FlowSettings;
var p = {};
var accountApplicationId = recordId;
p.accountApplicationId = accountApplicationId;
p.toStatus = "OCR in Progress";
ebs.callActionByName("myAccountApplication_BusinessWorkflow",
p, function(e) {
    var params = {};
    params.flowSettingsName = flowSettingsName;
    params.processorSettingsType = 'Onfido';
    params.processorSettingsName = 'Onfido_Settings'; //name
of the processor setting that you created
    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params, function(f) {
        var processorSettingsId =
f.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
processorSettingsId, recordId, false);
    });
});

```

4. Click the **Save and Close** button at the top right corner to save your digital journey.

AriadNext

The AriadNext automation processor facilitates customer verification processes by validating identity documents and automatically populating entity records in FintechOS applications with text extracted from document scans or photos. Customers can validate their identity by taking a picture of their ID card or uploading an existing one from their device.

All document scans that AriadNext handles are processed and transferred under strict, GDPR compliant, safety policies. Data from AriadNext can be deleted manually from the AriadNext account.

In addition, an applicant's identity is easily verified through the following options:

- Using the document liveness process where they are required to move the document in certain directions in order to ensure that the provided identity document is not a copy or a fake.
- Using the biometric liveness process where an action is required, such as moving the head to the right, in order to be validated as a real person. Along with this, the customer is compared to the photo from the provided identity document.

The identity data is then captured, examined, and validated based on the automation processor's [settings](#). Once the scanning process is complete, the customer's record populates with their personal information. The files resulted from the verification process (ID validation photo, document and biometric liveness recordings) can be downloaded and saved to the applicant's instance if needed.

AriadNext is ADR compliant and it follows the rules and regulations set in place by the Authority for the Digitization of Romania (ADR). These are meant to better facilitate video identity verification processes.

This solution benefits banking and insurance scenarios such as digital onboarding, Know Your Customer (KYC) processes, identity verification processes, and much more.

Key Features

- The ID validation process prevents identity fraud with document verification and facial biometrics.
- Data extracted from documents no longer needs to be added manually.
- Speeds up Know Your Customer (KYC) processes.
- ID cards and documents are captured and checked in real-time without any manual intervention. When this process is required, the manual validation can be added to comply with local legislation.

Applications

AriadNext can be implemented to simplify paper-driven financial or insurance processes, such as:

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes
- Claims handling
- Mortgage processing

Installing AriadNext

IMPORTANT!

Before proceeding with the installation, customers must set up an AriadNext account. The user name and password is obtained from the AriadNext platform. Additional information can be discussed with FintechOS sales representatives.

1 Install the SysPacks

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.

5. Download the **SySDigitalSolutionPackages v22.1.1000.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page,

2 Set up the AriadNext service subscription key

On the FintechOS Portal or B2C environment, open the **web.config** file in a text editor and add the following entries in the `<appSettings>` section, or in Vault:

In Vault

Key Path	Key Name	Value
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServicesAriadNextEndpoint	DCS web app endpoint URL provided by FintechOS.
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServicesAriadNextAppId	ID for the AriadNext service subscription.
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServicesAriadNextSubscriptionKey	Subscription key for the AriadNext service.

```

{
  "FTOSServicesAriadNextEndpoint": "URL"
  "FTOSServicesAriadNextAppId": "APP-ID"
  "FTOSServicesAriadNextSubscriptionKey":
  "SUBSCRIPTION-KEY"
}
```

In web.config

Key	Value
FTOSServicesAriadNextEndpoint	DCS web app endpoint URL provided by FintechOS.

Key	Value
FTOSServicesAriadNextAppId	ID for the AriadNext service subscription.
FTOSServicesAriadNextSubscriptionKey	Subscription key for the AriadNext service.

```
<add
key
="FTOSServicesAriadNextEndpoint" value="https://"/>
<add key="FTOSServicesAriadNextAppId" value="AppID"/>
<add
key
="FTOSServicesAriadNextSubscriptionKey"
value="SubscriptionKey"/>
```

Setting up the AriadNext Automation Processor

Follow the below steps when configuring the processor.

1 Create a generic processor settings group

The Automation Blocks automation processor must be hosted inside a generic processor settings group. A generic processor settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a generic processor settings group you wish to host your AriadNext automation processor, skip to "[2 Add the AriadNext automation processor to a generic processor settings group](#)" on the next page. Otherwise, follow the instructions below to create a new generic processor settings group:

1. In Innovation Studio, go to **Main Menu > Digital Journeys > Digital Flow Processing**. The **Flow Settings List** page opens.
2. In the **Flow Settings List** page, click the **Insert** button at the top right corner to add a new digital flow settings group. The **Add Digital Flow Settings** page opens.

3. In the **Add Digital Flow Settings** page, enter a **Name** for your digital flow settings group.
4. If you already have a digital journey set up where you wish to call the AriadNext automation processor, select it from the **Digital Journey** drop-down box.
5. Click the **Save and Close** button at the top right corner to save your flow settings group.

2 Add the AriadNext automation processor to a generic processor settings group

1. In Innovation Studio, go to **Main Menu > Digital Journeys > Digital Flow Processing**. The **Flow Settings List** page opens.
2. In the **Flow Settings List** page, double click the desired digital flow settings group.
3. In the **Processor Settings** section, click the **Insert** button.
4. Fill in the following fields:

Field	Description
Name	Enter a name for the processor settings.
Flow Settings	Leave the default value.
Digital Processor Type	Select AriadNext .
Settings	JSON code for the automation processor's settings. For details, see "AriadNext Settings" on the next page .
Mapping	JSON code for the automation processor's mappings. For details, see "AriadNext Workflow Statuses" on page 159 .

5. Click **Save and Close** at the top right corner of the screen.

NOTE

Customized configurations can be added to the Automation Blocks automation processor for a better user experience. These configuration can be related to colors, logo, wording, and so on. Make sure that the configurations are made before adding the processor to a Digital Journey.

AriadNext Settings

The AriadNext settings are defined in JSON format as key-value pairs. The following settings are available:

Setting	JSON Key	Description
UID File	FileUIDAttribute	The name of the source entity attribute on which the unique ID of the AriadNext file is saved. It holds a unique ID used for retrieving reports.
Workflow entity	SourceEntityPrimaryAttributeName	The name of the primary key attribute.
ADR Compliant	isAdrCompliant	The following options are available: <ul style="list-style-type: none"> • If true, the identity verification process is ADR compliant. • If false, the identity verification process is not ADR compliant.

Setting	JSON Key	Description
Configuration code	ConfigurationCode	<p>Configuration code obtained from configuration call which is done initially.</p> <div style="background-color: #e1eef6; padding: 10px; border-radius: 5px;"> <p>NOTE The configuration code is provided by FintechOS at implementation.</p> </div>
Webhook endpoint name	WebhookEndpointName	<p>Array containing the names of the notification endpoint obtained through the initial configuration call.</p> <div style="background-color: #e1eef6; padding: 10px; border-radius: 5px;"> <p>NOTE The webhook endpoint name is provided by FintechOS at implementation.</p> </div>
Features	Features	<p>The following options are available:</p> <ul style="list-style-type: none"> • DocumentLiveness: if true the document liveness is enabled in AriadNext SDK. • BiometricLiveness: if true the biometric liveness is enabled. This option cannot be set to true if DocumentLiveness is set to false.

Setting	JSON Key	Description
Workflow entity	SourceEntityName	<p>The entity associated with the business workflow (digital journey) that calls the OCR process. Needed only if the OCR process is used on an edit form (to alter an existing record) to update the workflow entity's business status after the scan (see BusinessStatusSuccess and BusinessStatusFail).</p> <p>If the OCR process is used on an insert form (to create a new record), this key is not needed.</p>
Redirect in case of success	MaskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful scan.</p> <ul style="list-style-type: none"> • entity – Entity name. • form – Form name of the above entity. • section – Optional parameter for the section name of the above form.

Setting	JSON Key	Description
Redirect in case of failure	MaskNextStepUrlFail	<p>Location in the user interface where the user is redirected after the maximum number of failed scan attempts (see "AriadNext Settings" on page 115).</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form.

Example

Sample JSON code for AriadNext settings

```

{
  "FileUIDAttribute": "fileUID",
  "SourceEntityPrimaryAttributeName":
  "FTOSTestProcessorsId",
  "SourceEntityName": "FTOS_Test_Processors",
  "isAdrCompliant": false,
  "ConfigurationCode": "fintechos_conf_1",
  "WebhookEndpointName": ["test_webhooks_endpoint"]
  "Features": {
    "DocumentLiveness": true,
    "BiometricLiveness": true
  },
  "maskNextStepURLSuccess": {
    "entity": "FTOS_Test_Processors",
    "form": "AriadNext",
    "section": "SuccessStep"
  },
  "maskNextStepURLFail": {
    "entity": "FTOS_Test_Processors",
    "form": "AriadNext",

```

```

        "section": "ErrorStep"
    },
    "UseLocalization": true
}

```

AriadNext Request Responses

The AriadNext OCR processor is composed of a client side script, four server side scripts, and a server side library. When the applicant's documents are validated, the call is made using the below methods. The OCR data is then saved in the entity based on the [Settings](#) configurations and the response is returned in JSON format.

IMPORTANT!

The data saved contains video files in mp4 format. In order to save them into the platform you must white list the mp4 extension to allow to save it into the Upload EBS folder of the HPFI platform.

Requests

- `getFile`: the result contains file reports.
- `getDocument`: the result contains full document reports.
- `getDocumentImage`: the result contains a base64 encoded image of the uploaded document

Requests Examples

Sample JSON code for AriadNext `getFile` response

```

{
  "response": {
    "uid": "file-4c06fafd-78af-4470-ad7d-0646b32a1c2e",
    "owner": "fintechos@ariadnext.com",
    "location": "default",
    "creationDate": "2022-03-29T09:19:29",
    "lastUpdateDate": "2022-03-29T09:19:33",
    "lastReportStatus": "ERROR",
    "lastAnalysisStatus": "OK",
  }
}

```

```

    "lastReport":{
      "checks":[
        {
          "fileUid":"file-4c06fafd-78af-4470-ad7d-
0646b32a1c2e",
          "identifier":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
          "title":"ID",
          "message":"The ID is not valid",
          "type":"UNKNOWN",
          "status":"ERROR",
          "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
          "subChecks":[
            {
              "dataReferences":null,
              "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
              "identifier":"BACKEND_ANALYSIS",
              "title":"Document analysis",
              "message":"The document has been
analysed",
              "type":"DOCUMENT_ACCEPTABILITY",
              "status":"OK",
              "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
              "subChecks":null
            },
            {
              "dataReferences":null,
              "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
              "identifier":"ID_ANALYSIS",
              "title":"ID analysis",
              "message":"The ID is not OK",
              "type":"UNKNOWN",
              "status":"ERROR",
              "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
              "subChecks":[
                {
                  "dataReferences":null,
                  "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
                  "identifier":"MODEL_RECOGNIZED",

```



```

        "title":"Document type
identification",
        "message":"Identified document",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",
        "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks":null
    },
    {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
        "identifiser":"DOC_SPECIMEN",
        "title":"Specimen",
        "message":"This document is not a
specimen",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",
        "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks":null
    },
    {
        "dataReferences":[
            {
                "givenValue":"19/07/2011",
                "expectedValue":null
            }
        ],
        "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
        "identifiser":"DOC_EXPIRATION_
DATE",
        "title":"Expiration of the
document",
        "message":"There are some
warnings concerning the document expiration date",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"ERROR",
        "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks":null
    },
    {

```

```

4470-ad7d-0646b32a1c2e",
  "dataReferences":null,
  "fileUid":"file-4c06fafd-78af-
detection",
  "identifier":"ID_FALSIFICATION",
  "title":"Falsification
falsified",
  "message":"Document is not
"type":"UNKNOWN",
  "status":"OK",
  "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
  "subChecks":[
    {
      "dataReferences":null,
      "fileUid":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
      "identifier":"DOC_
BLACKLISTED",
      "title":"Verification if
the document is blacklisted.",
      "message":"There was no
blacklist available for this document",
      "type":"DOCUMENT_VALIDITY",
      "status":"NONE",
      "documentUid":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
      "subChecks":null
    },
    {
      "dataReferences":[
        {
          "givenValue":"ROU",
          "expectedValue":null
        }
      ],
      "fileUid":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
      "identifier":"EMISSION_
COUNTRY",
      "title":"Document issue
country",
      "message":"The issue
country of the document is valid",
      "type":"DOCUMENT_VALIDITY",
      "status":"OK",

```

```

0fd6-46c2-9993-d7ec59811a19",
    "documentUId":"64cd6532-
    "subChecks":null
  },
  {
    "dataReferences":null,
    "fileUId":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
    "identifier":"EMISSION_
DATE",
    "title":"Document issue
date",
    "message":"The issue date
of the document has not been verified",
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUId":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":[
      {
        "givenValue":"ROU",
        "expectedValue":null
      }
    ],
    "fileUId":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
    "identifier":"DOC_
NATIONALITY",
    "title":"Holder
nationality",
    "message":"The holder
nationality is valid",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUId":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "fileUId":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",

```



```

    },
    {
      "givenValue": "BV183916<01328707198M110719110801192",
      "expectedValue": null
    }
  ],
  "fileUid": "file-4c06fafd-78af-4470-ad7d-0646b32a1c2e",
  "identifier": "MRZ_CHECKSUMS",
  "title": "MRZ checksums",
  "message": "All MRZ checksums are valid",
  "type": "DOCUMENT_VALIDITY",
  "status": "OK",
  "documentUid": "64cd6532-0fd6-46c2-9993-d7ec59811a19",
  "subChecks": null
},
{
  "dataReferences": null,
  "fileUid": "file-4c06fafd-78af-4470-ad7d-0646b32a1c2e",
  "identifier": "MRZ_EXPECTED_FOUND",
  "title": "Checking for the presence of a MRZ",
  "message": "MRZ expected by the model and found",
  "type": "DOCUMENT_VALIDITY",
  "status": "OK",
  "documentUid": "64cd6532-0fd6-46c2-9993-d7ec59811a19",
  "subChecks": null
},
{
  "dataReferences": null,
  "fileUid": "file-4c06fafd-78af-4470-ad7d-0646b32a1c2e",
  "identifier": "OCR_FIRSTNAMES",
  "title": "Consistency of holder's firstnames",
  "message": "The holder's firstnames have not been verified",

```

```

    "type": "DOCUMENT_VALIDITY",
    "status": "NONE",
    "documentUid": "64cd6532-
0fd6-46c2-9993-d7ec59811a19",
    "subChecks": null
  },
  {
    "dataReferences": null,
    "fileUid": "file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
    "identifier": "OCR_
LASTNAME",
    "title": "Consistency of
holder's lastname",
    "message": "The holder's
lastname has not been verified",
    "type": "DOCUMENT_VALIDITY",
    "status": "NONE",
    "documentUid": "64cd6532-
0fd6-46c2-9993-d7ec59811a19",
    "subChecks": null
  },
  {
    "dataReferences": null,
    "fileUid": "file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
    "identifier": "OCR_
BIRTHDATE",
    "title": "Consistency of
holder's birth date",
    "message": "The holder's
birthdate has not been verified",
    "type": "DOCUMENT_VALIDITY",
    "status": "NONE",
    "documentUid": "64cd6532-
0fd6-46c2-9993-d7ec59811a19",
    "subChecks": null
  },
  {
    "dataReferences": null,
    "fileUid": "file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
    "identifier": "OCR_DOCNUM",
    "title": "Consistency of the
document number",

```

```

number has not been verified",
                                "message":"The document
                                "type":"DOCUMENT_VALIDITY",
                                "status":"NONE",
                                "documentUid":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
                                "subChecks":null
                                },
                                {
                                "dataReferences":null,
                                "fileUid":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
                                "identifier":"OCR_
EXPIRATIONDATE",
                                "title":"Consistency of the
expiration date",
                                "message":"The expiration
date consistency has not been verified",
                                "type":"DOCUMENT_VALIDITY",
                                "status":"NONE",
                                "documentUid":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
                                "subChecks":null
                                },
                                {
                                "dataReferences":null,
                                "fileUid":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
                                "identifier":"OCR_
EMISSIONDATE",
                                "title":"Consistency of the
emit date",
                                "message":"The emit date
has not been verified",
                                "type":"DOCUMENT_VALIDITY",
                                "status":"NONE",
                                "documentUid":"64cd6532-
0fd6-46c2-9993-d7ec59811a19",
                                "subChecks":null
                                },
                                {
                                "dataReferences":null,
                                "fileUid":"file-4c06fafd-
78af-4470-ad7d-0646b32a1c2e",
                                "identifier":"OCR_
PERSONALNUM",

```

```

personal number",
number has not been verified",
0fd6-46c2-9993-d7ec59811a19",
},
{
78af-4470-ad7d-0646b32a1c2e",
ALIGNEMENT",
graphical format",
graphical format seems correct",
0fd6-46c2-9993-d7ec59811a19",
},
{
78af-4470-ad7d-0646b32a1c2e",
CLASSIFIER",
the consistency of the MRZ with the document model",
consistent with the document model",
0fd6-46c2-9993-d7ec59811a19",
},
{
78af-4470-ad7d-0646b32a1c2e",
"title":"Consistency of the
"message":"The personal
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-
"identifier":"MRZ_
"title":"Validity of MRZ
"message":"The MRZ
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-
"identifier":"MRZ_
"title":"Verification of
"message":"The MRZ is
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-

```



```

CONFORMITY",
compliance check",
photo is legit",
0fd6-46c2-9993-d7ec59811a19",
},
{
78af-4470-ad7d-0646b32a1c2e",
LOCATION",
presence and location of the photo",
detected at the location indicated in the document model",
0fd6-46c2-9993-d7ec59811a19",
},
{
78af-4470-ad7d-0646b32a1c2e",
SECURITY",
graphical security elements",
security elements have not been verified",
0fd6-46c2-9993-d7ec59811a19",
}
]
}

"identifier":"PHOTO_
"title":"Extensive photo
"message":"The detected
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-
"identifier":"PHOTO_
"title":"Check of the
"message":"A photo has been
detected at the location indicated in the document model",
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-
"identifier":"VISUAL_
"title":"Verification of
"message":"The graphical
security elements have not been verified",
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-
"subChecks":null
}
]
}

```

```

    ]
  }
]
},
"documents": [
  {
    "documentUid": "64cd6532-0fd6-46c2-9993-
d7ec59811a19",
    "customers": [
      "012ad73e-341f-4f1e-815d-dbb9ab82cff6"
    ]
  }
],
"clientData": {
  "reference": null
},
"referenceValues": null,
"customerIdentities": [
  {
    "uid": "012ad73e-341f-4f1e-815d-dbb9ab82cff6",
    "creationDate": "2022-03-29T09:19:33",
    "documentUids": [
      "64cd6532-0fd6-46c2-9993-d7ec59811a19"
    ],
    "addressData": null,
    "chequeData": null,
    "consumptionData": null,
    "financeData": null,
    "identityData": {
      "lastName": {
        "valueLabel": null,
        "label": "Last name",
        "value": "JOACA BINE"
      },
      "firstNames": {
        "label": "First name",
        "values": [
          "MIREL"
        ]
      }
    },
    "fullName": null,
    "birthDate": {
      "day": 19,
      "month": 7,

```

```

        "year":1987,
        "label":"Birth date",
        "value":"19/07/1987"
    },
    "birthDay":{
        "valueLabel":null,
        "label":"Birth day",
        "value":"19"
    },
    "birthDepartment":null,
    "birthMonth":{
        "valueLabel":null,
        "label":"Birth month",
        "value":"7"
    },
    "birthYear":{
        "valueLabel":null,
        "label":"Birth year",
        "value":"1987"
    },
    "birthPlace":null,
    "gender":{
        "valueLabel":"Male",
        "label":"Gender",
        "value":"M"
    },
    "nationality":{
        "valueLabel":null,
        "label":"Nationality",
        "value":"ROU"
    },
    "faceUrl":null,
    "nationalRegistrationNumber":null,
    "ssn":null,
    "usageName":null
    },
    "jobData":null,
    "role":null,
    "vehicleData":null
    }
],
"uid":"a936cbf5-861f-4b9f-a36b-90a5e08195ec",
"generationDate":"2022-03-29T09:19:33",
"globalStatus":"ERROR"
},

```

```

"documents":[
  {
    "uid":"64cd6532-0fd6-46c2-9993-d7ec59811a19",
    "type":"ID",
    "subType":"ID"
  }
],
"reports":[
  {
    "uid":"a936cbf5-861f-4b9f-a36b-90a5e08195ec",
    "generationDate":"2022-03-29T09:19:33",
    "globalStatus":"ERROR"
  }
],
"validity":"NOT_VALIDATED",
"state":"INITIAL",
"tags":[
  "sdk-web"
],
"type":null,
"comments":null,
"isSuccess":true,
"errorMessage":null
},
"lastReport":{
  "checks":[
    {
      "fileUid":"file-4c06fafd-78af-4470-ad7d-
0646b32a1c2e",
      "identifier":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
      "title":"ID",
      "message":"The ID is not valid",
      "type":"UNKNOWN",
      "status":"ERROR",
      "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
      "subChecks":[
        {
          "dataReferences":null,
          "fileUid":"file-4c06fafd-78af-4470-ad7d-
0646b32a1c2e",
          "identifier":"BACKEND_ANALYSIS",
          "title":"Document analysis",

```

```

    analysed",
    "message":"The document has been
    "type":"DOCUMENT_ACCEPTABILITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "fileUid":"file-4c06fafd-78af-4470-ad7d-
0646b32a1c2e",
    "identifier":"ID_ANALYSIS",
    "title":"ID analysis",
    "message":"The ID is not OK",
    "type":"UNKNOWN",
    "status":"ERROR",
    "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
    "subChecks":[
      {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
        "identifier":"MODEL_RECOGNIZED",
        "title":"Document type
identification",
        "message":"Identified document",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",
        "documentUid":"64cd6532-0fd6-46c2-
9993-d7ec59811a19",
        "subChecks":null
      },
      {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
        "identifier":"DOC_SPECIMEN",
        "title":"Specimen",
        "message":"This document is not a
specimen",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",

```

```

9993-d7ec59811a19",
    "documentUid":"64cd6532-0fd6-46c2-
ad7d-0646b32a1c2e",
    "subChecks":null
  },
  {
    "dataReferences":[
      {
        "givenValue":"19/07/2011",
        "expectedValue":null
      }
    ],
    "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
    "identifier":"DOC_EXPIRATION_DATE",
    "title":"Expiration of the
document",
    "message":"There are some warnings
concerning the document expiration date",
    "type":"DOCUMENT_ACCEPTABILITY",
    "status":"ERROR",
    "documentUid":"64cd6532-0fd6-46c2-
9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "fileUid":"file-4c06fafd-78af-4470-
ad7d-0646b32a1c2e",
    "identifier":"ID_FALSIFICATION",
    "title":"Falsification detection",
    "message":"Document is not
falsified",
    "type":"UNKNOWN",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-46c2-
9993-d7ec59811a19",
    "subChecks":[
      {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
        "identifier":"DOC_
BLACKLISTED",
        "title":"Verification if the
document is blacklisted.",

```

```

        "message": "There was no
blacklist available for this document",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": [
            {
                "givenValue": "ROU",
                "expectedValue": null
            }
        ],
        "fileUid": "file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
        "identifier": "EMISSION_
COUNTRY",
        "title": "Document issue
country",
        "message": "The issue country
of the document is valid",
        "type": "DOCUMENT_VALIDITY",
        "status": "OK",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": null,
        "fileUid": "file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
        "identifier": "EMISSION_DATE",
        "title": "Document issue date",
        "message": "The issue date of
the document has not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": [

```



```

FOUND",
presence of a MRZ",
model and found",
46c2-9993-d7ec59811a19",
},
{
"identifier":"MRZ_EXPECTED_",
"title":"Checking for the
"message":"MRZ expected by the
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-0fd6-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
"identifier":"OCR_FIRSTNAMES",
"title":"Consistency of
holder's firstnames",
"message":"The holder's
firstnames have not been verified",
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
"identifier":"OCR_LASTNAME",
"title":"Consistency of
holder's lastname",
"message":"The holder's
lastname has not been verified",
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",

```

```

        "holder's birth date",
        "birthdate has not been verified",
        "46c2-9993-d7ec59811a19",
        "4470-ad7d-0646b32a1c2e",
        "document number",
        "has not been verified",
        "46c2-9993-d7ec59811a19",
        "4470-ad7d-0646b32a1c2e",
        "EXPIRATIONDATE",
        "expiration date",
        "consistency has not been verified",
        "46c2-9993-d7ec59811a19",
        "4470-ad7d-0646b32a1c2e",
        "identifier":"OCR_BIRTHDATE",
        "title":"Consistency of",
        "message":"The holder's",
        "type":"DOCUMENT_VALIDITY",
        "status":"NONE",
        "documentUid":"64cd6532-0fd6-",
        "subChecks":null
    },
    {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-",
        "identifier":"OCR_DOCNUM",
        "title":"Consistency of the",
        "message":"The document number",
        "type":"DOCUMENT_VALIDITY",
        "status":"NONE",
        "documentUid":"64cd6532-0fd6-",
        "subChecks":null
    },
    {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-",
        "identifier":"OCR_",
        "title":"Consistency of the",
        "message":"The expiration date",
        "type":"DOCUMENT_VALIDITY",
        "status":"NONE",
        "documentUid":"64cd6532-0fd6-",
        "subChecks":null
    },
    {
        "dataReferences":null,
        "fileUid":"file-4c06fafd-78af-",

```

```

EMISSIONDATE",
emit date",
not been verified",
46c2-9993-d7ec59811a19",
},
{
4470-ad7d-0646b32a1c2e",
PERSONALNUM",
personal number",
has not been verified",
46c2-9993-d7ec59811a19",
},
{
4470-ad7d-0646b32a1c2e",
graphical format",
format seems correct",
46c2-9993-d7ec59811a19",
},
{
"identifier":"OCR_
"title":"Consistency of the
"message":"The emit date has
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-0fd6-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-78af-
"identifier":"OCR_
"title":"Consistency of the
"message":"The personal number
"type":"DOCUMENT_VALIDITY",
"status":"NONE",
"documentUid":"64cd6532-0fd6-
"subChecks":null
},
{
"dataReferences":null,
"fileUid":"file-4c06fafd-78af-
"identifier":"MRZ_ALIGNMENT",
"title":"Validity of MRZ
"message":"The MRZ graphical
"type":"DOCUMENT_VALIDITY",
"status":"OK",
"documentUid":"64cd6532-0fd6-
"subChecks":null
},
{
"dataReferences":null,

```

```

4470-ad7d-0646b32a1c2e",
    "fileUid":"file-4c06fafd-78af-
consistency of the MRZ with the document model",
    "identifier":"MRZ_CLASSIFIER",
    "title":"Verification of the
message":"The MRZ is
consistent with the document model",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
},
{
    "dataReferences":null,
    "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
    "identifier":"PHOTO_
CONFORMITY",
    "title":"Extensive photo
compliance check",
    "message":"The detected photo
is legit",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
},
{
    "dataReferences":null,
    "fileUid":"file-4c06fafd-78af-
4470-ad7d-0646b32a1c2e",
    "identifier":"PHOTO_LOCATION",
    "title":"Check of the presence
and location of the photo",
    "message":"A photo has been
detected at the location indicated in the document model",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
},
{
    "dataReferences":null,

```

```

4470-ad7d-0646b32a1c2e",
    "fileUid":"file-4c06fafd-78af-
SECURITY",
    "identifier":"VISUAL_
graphical security elements",
    "title":"Verification of
security elements have not been verified",
    "message":"The graphical
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  }
]
}
],
"documents":[
{
  "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
  "customers":[
    "012ad73e-341f-4f1e-815d-dbb9ab82cff6"
  ]
},
],
"clientData":{
  "reference":null
},
"referenceValues":null,
"customerIdentities":[
{
  "uid":"012ad73e-341f-4f1e-815d-dbb9ab82cff6",
  "creationDate":"2022-03-29T09:19:33",
  "documentUids":[
    "64cd6532-0fd6-46c2-9993-d7ec59811a19"
  ],
  "addressData":null,
  "chequeData":null,
  "consumptionData":null,
  "financeData":null,

```

```
"identityData":{
  "lastName":{
    "valueLabel":null,
    "label":"Last name",
    "value":"JOACA BINE"
  },
  "firstNames":{
    "label":"First name",
    "values":[
      "MIREL"
    ]
  },
  "fullName":null,
  "birthDate":{
    "day":19,
    "month":7,
    "year":1987,
    "label":"Birth date",
    "value":"19/07/1987"
  },
  "birthDay":{
    "valueLabel":null,
    "label":"Birth day",
    "value":"19"
  },
  "birthDepartment":null,
  "birthMonth":{
    "valueLabel":null,
    "label":"Birth month",
    "value":"7"
  },
  "birthYear":{
    "valueLabel":null,
    "label":"Birth year",
    "value":"1987"
  },
  "birthPlace":null,
  "gender":{
    "valueLabel":"Male",
    "label":"Gender",
    "value":"M"
  },
  "nationality":{
    "valueLabel":null,
    "label":"Nationality",
```

```

        "value": "ROU"
      },
      "faceUrl": null,
      "nationalRegistrationNumber": null,
      "ssn": null,
      "usageName": null
    },
    "jobData": null,
    "role": null,
    "vehicleData": null
  }
],
"uid": "a936cbf5-861f-4b9f-a36b-90a5e08195ec",
"generationDate": "2022-03-29T09:19:33",
"globalStatus": "ERROR"
}
}

```

Sample JSON code for AriadNext getDocument response

```

{
  "isSuccess": true,
  "errorMessage": "",
  "document": {
    "IsSuccess": true,
    "Response": {
      "uid": "64cd6532-0fd6-46c2-9993-d7ec59811a19",
      "owner": "fintechos@ariadnext.com",
      "location": "default",
      "type": "ID",
      "subType": "ID",
      "classId": "ROU_ID_0111",
      "prettyName": "Romania National ID Card 1997",
      "lastReport": {
        "checks": [
          {
            "identifier": "BACKEND_ANALYSIS",
            "title": "Document analysis",
            "message": "The document has been analysed",
            "type": "DOCUMENT_ACCEPTABILITY",
            "status": "OK",

```



```

    "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
    "subChecks":null
  },
  {
    "identifier":"ID_ANALYSIS",
    "title":"ID analysis",
    "message":"The ID is not OK",
    "type":"UNKNOWN",
    "status":"ERROR",
    "documentUid":"64cd6532-0fd6-46c2-9993-
d7ec59811a19",
    "subChecks":[
      {
        "dataReferences":null,
        "identifier":"MODEL_RECOGNIZED",
        "title":"Document type
identification",
        "message":"Identified document",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",
        "documentUid":"64cd6532-0fd6-46c2-
9993-d7ec59811a19",
        "subChecks":null
      },
      {
        "dataReferences":null,
        "identifier":"DOC_SPECIMEN",
        "title":"Specimen",
        "message":"This document is not a
specimen",
        "type":"DOCUMENT_ACCEPTABILITY",
        "status":"OK",
        "documentUid":"64cd6532-0fd6-46c2-
9993-d7ec59811a19",
        "subChecks":null
      },
      {
        "dataReferences":[
          {
            "givenValue":"19/07/2011",
            "expectedValue":null
          }
        ],
        "identifier":"DOC_EXPIRATION_DATE",

```

```

        "title": "Expiration of the
document",
        "message": "There are some warnings
concerning the document expiration date",
        "type": "DOCUMENT_ACCEPTABILITY",
        "status": "ERROR",
        "documentUid": "64cd6532-0fd6-46c2-
9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": null,
        "identifier": "ID_FALSIFICATION",
        "title": "Falsification detection",
        "message": "Document is not
falsified",
        "type": "UNKNOWN",
        "status": "OK",
        "documentUid": "64cd6532-0fd6-46c2-
9993-d7ec59811a19",
        "subChecks": [
            {
                "dataReferences": null,
                "identifier": "DOC_
BLACKLISTED",
                "title": "Verification if the
document is blacklisted.",
                "message": "There was no
blacklist available for this document",
                "type": "DOCUMENT_VALIDITY",
                "status": "NONE",
                "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
                "subChecks": null
            }
        ],
        "dataReferences": [
            {
                "givenValue": "ROU",
                "expectedValue": null
            }
        ],
        "identifier": "EMISSION_
COUNTRY",
    }

```

```

country",
of the document is valid",
46c2-9993-d7ec59811a19",
    "title":"Document issue
    "message":"The issue country
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"EMISSION_DATE",
    "title":"Document issue date",
    "message":"The issue date of
the document has not been verified",
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":[
      {
        "givenValue":"ROU",
        "expectedValue":null
      }
    ],
    "identifier":"DOC_
NATIONALITY",
    "title":"Holder nationality",
    "message":"The holder
nationality is valid",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"VALIDITY_
PERIOD",

```



```

        "expectedValue":null
      }
    ],
    "identifier":"MRZ_CHECKSUMS",
    "title":"MRZ checksums",
    "message":"All MRZ checksums
are valid",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"MRZ_EXPECTED_
FOUND",
    "title":"Checking for the
presence of a MRZ",
    "message":"MRZ expected by the
model and found",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"OCR_FIRSTNAMES",
    "title":"Consistency of
holder's firstnames",
    "message":"The holder's
firstnames have not been verified",
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"OCR_LASTNAME",
    "title":"Consistency of
holder's lastname",

```

```

        "message": "The holder's
lastname has not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": null,
        "identifier": "OCR_BIRTHDATE",
        "title": "Consistency of
holder's birth date",
        "message": "The holder's
birthdate has not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": null,
        "identifier": "OCR_DOCNUM",
        "title": "Consistency of the
document number",
        "message": "The document number
has not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    },
    {
        "dataReferences": null,
        "identifier": "OCR_
EXPIRATIONDATE",
        "title": "Consistency of the
expiration date",
        "message": "The expiration date
consistency has not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
    }
}

```

```

46c2-9993-d7ec59811a19",
    "documentUid":"64cd6532-0fd6-
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"OCR_
EMISSIONDATE",
    "title":"Consistency of the
emit date",
    "message":"The emit date has
not been verified",
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"OCR_
PERSONALNUM",
    "title":"Consistency of the
personal number",
    "message":"The personal number
has not been verified",
    "type":"DOCUMENT_VALIDITY",
    "status":"NONE",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  {
    "dataReferences":null,
    "identifier":"MRZ_ALIGNMENT",
    "title":"Validity of MRZ
graphical format",
    "message":"The MRZ graphical
format seems correct",
    "type":"DOCUMENT_VALIDITY",
    "status":"OK",
    "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
    "subChecks":null
  },
  },

```

```

        {
            "dataReferences":null,
            "identifier":"MRZ_CLASSIFIER",
            "title":"Verification of the
consistency of the MRZ with the document model",
            "message":"The MRZ is
consistent with the document model",
            "type":"DOCUMENT_VALIDITY",
            "status":"OK",
            "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
            "subChecks":null
        },
        {
            "dataReferences":null,
            "identifier":"PHOTO_
CONFORMITY",
            "title":"Extensive photo
compliance check",
            "message":"The detected photo
is legit",
            "type":"DOCUMENT_VALIDITY",
            "status":"OK",
            "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
            "subChecks":null
        },
        {
            "dataReferences":null,
            "identifier":"PHOTO_LOCATION",
            "title":"Check of the presence
and location of the photo",
            "message":"A photo has been
detected at the location indicated in the document model",
            "type":"DOCUMENT_VALIDITY",
            "status":"OK",
            "documentUid":"64cd6532-0fd6-
46c2-9993-d7ec59811a19",
            "subChecks":null
        },
        {
            "dataReferences":null,
            "identifier":"VISUAL_
SECURITY",

```



```

        "title": "Verification of
graphical security elements",
        "message": "The graphical
security elements have not been verified",
        "type": "DOCUMENT_VALIDITY",
        "status": "NONE",
        "documentUid": "64cd6532-0fd6-
46c2-9993-d7ec59811a19",
        "subChecks": null
    }
}
]
}
],
"issuance": {
    "issueDate": null,
    "issueDay": null,
    "issueMonth": null,
    "issueYear": null,
    "issuingAuthority": null,
    "issuingCountry": {
        "valueLabel": null,
        "label": "Issuing country",
        "value": "ROU"
    }
},
"info": {
    "cardAccessNumber": null,
    "documentNumber": {
        "valueLabel": null,
        "label": "Document number",
        "value": "BV183916"
    },
    "personalNumber": {
        "valueLabel": null,
        "label": "Personal number",
        "value": "1870719080119"
    },
    "documentType": {
        "valueLabel": null,
        "label": "Document type",
        "value": "ID"
    },
    "expirationDate": {
        "day": 19,

```

```

        "month":7,
        "year":2011,
        "label":"Expiration date",
        "value":"19/07/2011"
    },
    "expirationDay":{
        "valueLabel":null,
        "label":"Expiration day",
        "value":"19"
    },
    "expirationMonth":{
        "valueLabel":null,
        "label":"Expiration month",
        "value":"7"
    },
    "expirationYear":{
        "valueLabel":null,
        "label":"Expiration year",
        "value":"2011"
    },
    "readExpirationDate":{
        "day":19,
        "month":7,
        "year":2011,
        "label":"Read expiration date",
        "value":"19/07/2011"
    },
    "sidesIssue":null
},
"persons":[
    {
        "addressData":null,
        "chequeData":null,
        "consumptionData":null,
        "financeData":null,
        "identityData":{
            "lastName":{
                "valueLabel":null,
                "label":"Last name",
                "value":"JOACA BINE"
            },
            "firstNames":{
                "label":"First name",
                "values":[
                    "MIREL"
                ]
            }
        }
    }
]

```

```

    ]
  },
  "fullName":null,
  "birthDate":{
    "day":19,
    "month":7,
    "year":1987,
    "label":"Birth date",
    "value":"19/07/1987"
  },
  "birthDay":{
    "valueLabel":null,
    "label":"Birth day",
    "value":"19"
  },
  "birthDepartment":null,
  "birthMonth":{
    "valueLabel":null,
    "label":"Birth month",
    "value":"7"
  },
  "birthYear":{
    "valueLabel":null,
    "label":"Birth year",
    "value":"1987"
  },
  "birthPlace":null,
  "gender":{
    "valueLabel":"Male",
    "label":"Gender",
    "value":"M"
  },
  "nationality":{
    "valueLabel":null,
    "label":"Nationality",
    "value":"ROU"
  },
  "faceUrl":null,
  "nationalRegistrationNumber":null,
  "ssn":null,
  "usageName":null
},
"jobData":null,
"role":{
  "valueLabel":"Holder",

```

```

        "label": "Role",
        "value": "ID"
      },
      "vehicleData": null
    }
  ],
  "backendResultId": "215750",
  "uid": "8fece2e5-f51c-4c5b-a429-6afaab8d4e98",
  "generationDate": "2022-03-29T09:19:33",
  "globalStatus": "ERROR"
},
"lastAnalysisStatus": "OK",
"reports": [
  {
    "uid": "8fece2e5-f51c-4c5b-a429-6afaab8d4e98",
    "generationDate": "2022-03-29T09:19:33",
    "globalStatus": "ERROR"
  }
],
"images": [
  {
    "uid": "29abf611-8f34-42b1-8e6f-c8fc5d8d59a0",
    "source": "ORIGINAL",
    "documentPart": "RECTO",
    "type": "DL",
    "origin": "ORIGINAL",
    "sourceImageUid": null
  },
  {
    "uid": "254115be-dcf9-44b2-9041-6bfcd3469415",
    "source": "CROPPED",
    "documentPart": "RECTO",
    "type": "DL",
    "origin": "CROPPED_RECTO",
    "sourceImageUid": "29abf611-8f34-42b1-8e6f-
c8fc5d8d59a0"
  },
  {
    "uid": "1ff21068-2dca-443a-94ef-bee57274c38f",
    "source": "CROPPED",
    "documentPart": "OTHER",
    "type": "DL",
    "origin": "CROPPED_FACE",
    "sourceImageUid": "254115be-dcf9-44b2-9041-
6bfcd3469415"
  }
]

```

```

    }
  ],
  "creationDate": "2022-03-29T09:19:29",
  "lastUpdateDate": "2022-03-29T09:19:33",
  "extraData": {
    "emrtdData": {
      "deviceCompatible": false,
      "errorMessage": null,
      "nfcAuthorized": false,
      "sessionStatus": null
    },
    "deviceInfo": {
      "model": null,
      "nfcAvailable": false,
      "osVersion": null,
      "skd": null,
      "skdEmrtdActivated": false,
      "sdkVersion": null,
      "sdkVideoScanActivated": false
    },
    "videoScanData": {
      "sessionRectoStatus": null,
      "sessionVersoStatus": null,
      "errorMessage": null
    },
    "livenessData": {
      "livenessReadiness": "LIVENESS_READY"
    },
    "envelopeKeys": [
    ]
  },
  "isSuccess": true,
  "errorMessage": null
},
"statusCode": 200
}

```

Requests - ADR

- `getIdentityProofing`: the result contains an object with `identityProofing` information.
- `getIdentityProofingFile`: the result contains an object of the uploaded document.

Requests Examples - ADR

Sample JSON code for AriadNext `getIdentityProofing` response

NOTE

`getIdentityProofingId` is a GET type request that has the `identityProofingId` parameter in the link.

```
{
  "id": "163f3869-cd49-4b49-8a01-f34d3ebc7bb4",
  "createdDate": "7/14/2022 7:37:04 AM",
  "workflowStatus": "CREATED",
  "error": null,
  "captureMode": null,
  "analysis": null,
  "isSuccess": true,
  "errorMessage": null
}
```

Sample JSON code for AriadNext `getIdentityProofingFile` response

NOTE

`getIdentityProofingFile` is a GET type request that has two parameters in the link: `identityProofingId` and an ID for the file that you want to retrieve.

```
{
  "imageArray": "{BASE64_ENCODED_IMAGE}",
  "isSuccess": true,
}
```

```

    "errorMessage": null
  }

```

AriadNext Workflow Statuses

When going through the Automation Blocks identity verification flow, the following verdict statuses are returned:

- Verdict Success: WorkflowStatus = VERDICT_AVAILABLE et verdict = SUCCESS
Returned when the data extracted from the document and reviewed by manual operators can be retrieved.
- Verdict Failure: WorkflowStatus = VERDICT_AVAILABLE and verdict = FAILURE
Returned when an error code is sent.

For a SUCCESS verdict status, the owner and document data, and the captured and extracted images are available for manual review.

If the status verdict displays FAILURE, an error is returned notifying that the verification process could not be completed. In this case, there are two types of errors returned:

Error Type	Description
FRAUDULENT_IDENTITY_SUSPICION	The person's identity could not be validated and fraud is suspected.

Error Type	Description
TECHNICAL_REASON	<p>Depending on the error source, the following codes are displayed, preventing the verification process to be complete:</p> <ul style="list-style-type: none"> • DEVICE_ISSUE: issue with the device during the identity verification flow • DOCUMENT_ISSUE: issue with the document during the verification flow • NETWORK_ISSUE: issue with the network during the verification flow • TECHNICAL_ISSUE: generic code for technical issues that appear during the verification flow • USER_ISSUE: issue with the user during the verification flow • USER_CANCEL: the user canceled the process and the flow could not be completed • FACE_TECHNICAL_ISSUE: issue with the live capture of the user's face during the verification flow • OTHER_QA: generic code for other quality issues that appear on captures during the verification flow

Adding AriadNext to a Digital Journey

1. Open the digital journey in Innovation Studio.
2. Make sure that the form you want to populate includes a button to call the AriadNext automation processor.
3. Click the **Advanced** tab.

4. Click the **After Events** subtab.
5. Add the following code in the **After Events** window:

```
var dfpHelper = ebs.importClientScript('FTOS.DFP');
var p = {};
p.flowSettingsName = formData.FlowSettings;
p.processorSettingsType = 'AriadNext';
p.processorSettingsName='AriadNext_Settings'; //the name of
the processor settings that you created.
var recordId = formData.id;
var rec2= ebs.getCurrentEntityId();
ebs.callActionByName("FTOS_DFP_FlowProcessorSettingsByType",
p, function(e)
{
    var processorSettingsId =
e.UIResult.Data.ProcessorSettingsId;
    dfpHelper.loadComponent("FTOS_DFP_AriadNext",
processorSettingsId, ebs.getCurrentEntityId(), false);
});
```

6. Click the **Save and Close** button at the top right corner to save the digital journey.

Face Recognition and Video

Face recognition technology is used to automatically validate an individual's identity from an image or video source. You can use this functionality to verify if a person from a digital image (selfie) or a video, matches the photo from their identity card. A face capture process is essential in Know Your Customer (KYC) or onboarding processes, to validate the customer's identity and prevent fraud.

Operators can also use video streaming to start a live video with their customers, to complete the identity verification process and provide assistance if needed.

Along with face recognition and video streaming, FintechOS offers Co-browsing technology allowing operators and customers to connect and chat in real time.

Integrate the below FintechOS components in your digital journeys to further enhance their usability.

Face Recognition

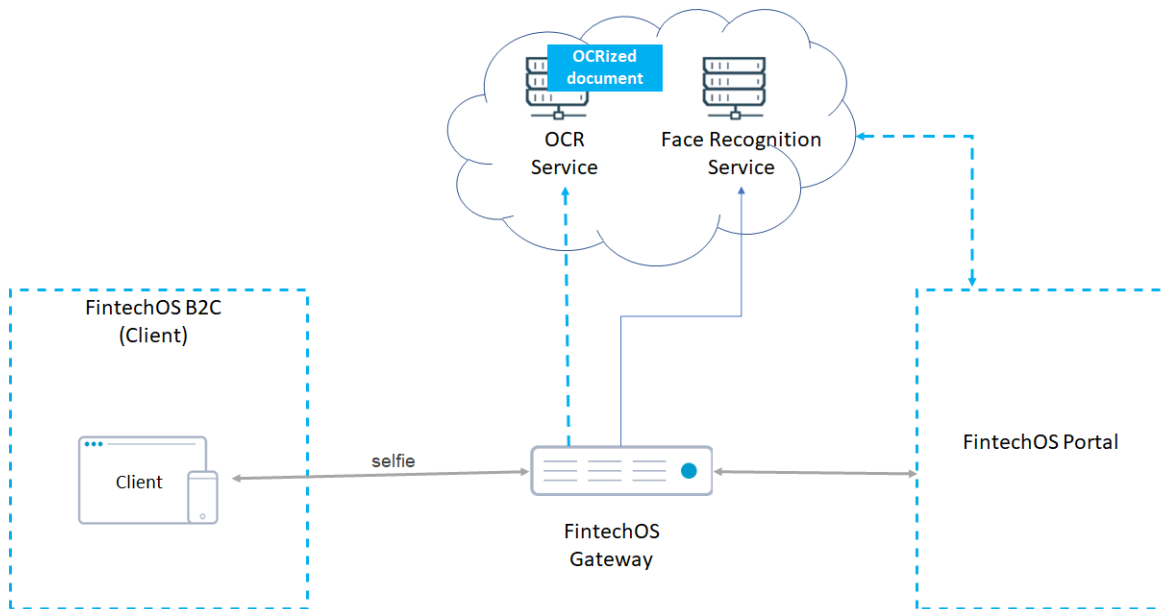
A facial recognition system is a technology capable of identifying or verifying a person from a digital image by comparing the given picture with one existing in the database.

The Face Recognition automation processor is using Machine Learning to compare ID/ Driving License or any other picture that was uploaded during the OCR process, with the selfie to certify that they belong to the same person, using the AI-enabled capabilities of the FintechOS platform.

To use the Face Recognition automation processor capabilities, the user should first use the Computer Vision automation processor to enable the clients uploading an identity document that will be used during the face recognition process.

The face recognition flow:

1. The client takes a selfie.
2. The FintechOS Gateway identifies the client's PID and sends a register Face Recognition request to the Face Recognition Service.
3. Based on the PID received from the FintechOS Gateway, the Face Recognition Service compares the client's image file (selfie) with the image from the identity document the client uploaded during the OCR process.
4. As a result of the comparison, the Face Recognition Service returns the confidence score of the face recognition comparison.



Once the Face Recognition is completed (i.e. the confidence score meets the minimum value set during the processor's configuration, see "[Face Recognition Mappings](#)" on page 180), you can provide the customers with various options:

- start a video call with one of the bank's consultants ([Video Streaming processor](#))
- sign directly all associated documents ([eSign processor](#))
- choose to be contacted later by one of the bank's consultants.

Face Recognition Processor Features

- real-time image recognition
- liveness detection to validate an individual's identity
- reduced bandwidth and time-to-decision
- anti-spoofing mechanisms
- increased privacy and reliability

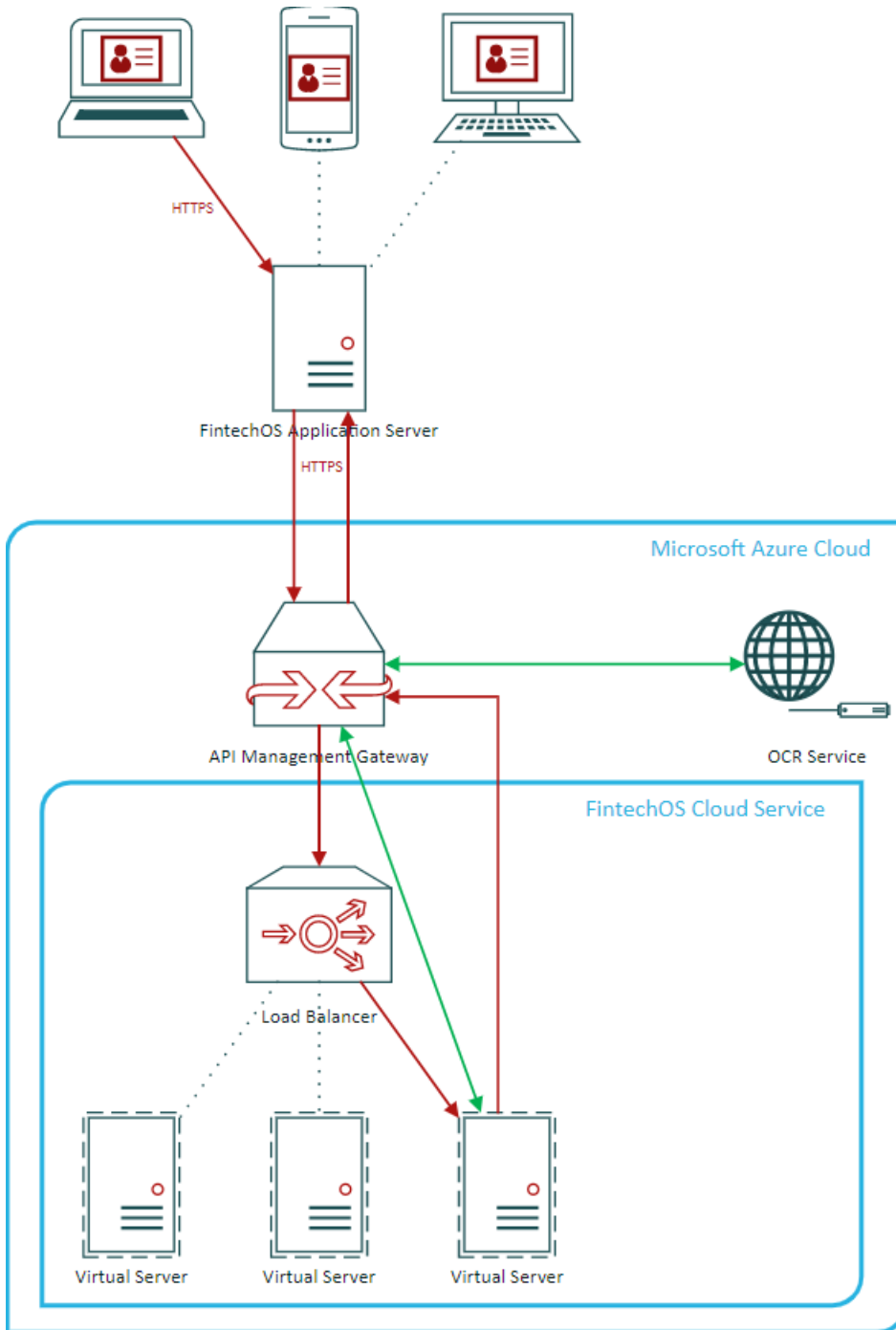
Applications

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes
- Claims handling
- Mortgage processing.

Data Security

All document scans are processed and transferred under strict, GDPR compliant, safety policies. Through the use of a secure protocol, files are sent to the Azure Cognitive Services Face service, which renders a score, that is finally returned to the FintechOS application. Data is safe under strict supervision as it follows a clear path from the sender to the FintechOS cloud.

Data Flow



1. The end-user sends the document scan and selfie to the FintechOS application server over secure communication channels (HTTPS encrypted messages, including the HTTP headers and request/response data).
2. The application server sends the files to the FintechOS cloud service via the Azure API Management gateway, also using HTTPS. The API Management gateway ensures secure communication and provides identity and access management to the FintechOS cloud service.
3. The files arrive at the FintechOS cloud service (hosted on a private load-balanced cluster of virtual machines in the Azure cloud). The virtual machines are managed by FintechOS and can be accessed only using the API Management services (no Internet access is allowed to any virtual machine or load-balanced cluster).
4. The FintechOS cloud service processes the ID document (each ID card field is delimited), and forwards the ID picture and selfie for face recognition to the Azure Cognitive Services Face service (also hosted on the Azure cloud).
5. The face recognition service returns the face recognition confidence score to the FintechOS cloud service, which sends the information back to the FintechOS application via the API Management gateway.

IMPORTANT!

No data is stored in a cloud. All processed information is immediately deleted.

Location

The API Management gateway, the FintechOS cloud service (load balancer and virtual machines), and the face recognition service are provisioned using the Microsoft Azure cloud service in the Western Europe data center (Amsterdam, Netherlands) with fail-over backup services on the Northern Europe data center (Dublin, Ireland). No data leaves the European Union in transit or at rest.

Based on customer requirements, similar services may be provisioned in the future in other regions.

Compliance

For cloud services compliance information, see:

- [Overview of Microsoft Azure compliance](#)
- [Microsoft Services Trust Portal](#).

Installing Face Recognition

1 Install Server Configuration

First, make sure you have the right application dependencies installed and configured. Go to the physical location of the site (Portal or B2C Portal) and follow the instructions below:

1. Portal/ B2C Portal web.config <appSettings> section, configuration

If you have a generic key for all the services, add the following keys in web.config - <appSettings>:

```
<add key="FTOSServicesEndpoint" value="get-the-url-from-portal"/>
<add key="FTOSServicesAppId" value="get-the-key-from-portal"/>
```

If you have different subscription keys for each of the services you must add the following keys in web.config - <appSettings>:

- **OCR Microsoft Azure provider**

```
<add key="FTOSServicesOCR2Endpoint" value="get-the-url-from-portal"/>
<add key="FTOSServicesOCR2AppId" value="get-the-key-from-portal"/>
```

- **Face Recognition**

```
<add key="FTOSServicesFaceEndpoint" value="get-the-url-
from-portal"/>
<add key="FTOSServicesFaceAppId" value="get-the-key-
from-portal"/>
```

- **Video**

```
<add key="FTOSServicesVideoFaceEndpoint" value="get-the-
url-from-portal"/>
<add key="FTOSServicesVideoAppId" value="get-the-key-
from-portal"/>
```

- **FaceRecognition With Liveness**

```
<add key="FTOSServicesLivenessEndpoint" value="get-the-
url-from-portal"/>
<add key="FTOSServicesLivenessAppId" value="get-the-key-
from-portal"/>
```

2. Custom Folder

For each installed Portal (B2C or BackOffice) that will use the cognitive video components, copy the Custom For FintechOS Portal\custom and Custom For FintechOS Portal\custom-on-demand folders from the Cognitive Processor pack in the application - portal web application root folder.

3. Copy to UploadEBS

Add the following images to the Upload EBS folder <portal_EBS_folder>:

```
<syspack_file_path>\10 Cognitive Processor Operator -
Vxx.x.xxxx \CopyToUploadEBS\emptyOCR.jpg

<syspack_file_path>\10 Cognitive Processor Operator -
Vxx.x.xxxx\CopyToUploadEBS\emptyPhoto.png
```

4. In IIS

The following MIME Types must be added on the sites where the SDK was previously added:

- . with type text/xml
- .data with type text/xml
- .wasm with type application/wasm

2 Install Application Configuration

Log into Innovation Studio, Developer profile, and follow the instructions below:

1. Import Packs

Go to **DevOps > Deployment Packages**, and import the received packs. The Cognitive Processor pack contains the services configuration model and scripts for OCR, Face Recognition and Video.

2. Import data

Go to **Evolutive Data Core > Data Import Template**. Select the template and import using the xlsx data in the packages:

- Import_01_FTOS_DFP_FlowSettings:

- Import_01_FTOS_DFP_FlowSettings.xlsx

- Import_FTOS_DFP_ProcessorSettings:

- Import_02_FTOS_DFP_ProcessorSettings.xlsx

3. Custom Setting

Go to **Admin > Settings**. Check the **Use Custom Styles** box. Click **Save and close**.

Log into the Portal.

NOTE For the OCR, Face Recognition, and Video components to work, the sites must run on HTTPS.

4. OCR Configuration

For configuring the OCR, select **Digital Flow Processing > Flow Settings**.

You have 2 flows as examples for OCR Romanian IDs: - " 2 OCR Flow Setting Example ". Here you can find examples of usage, and these processor settings are used by the "Example OCR Pack" from the package. It's an example for 2 step OCR (if your flow has 2 OCR steps in it).

OCR_1 - Reads an area of the ID.

OCR_2 - Reads a different area of the ID.

NOTE You can copy the examples and modify them.

In the **Example Flow Setting**, you can find a full IdRom setting, and mapping examples for edit and insert forms. For other types of OCR: Driving Licence, Passport, IdBG, the examples are the same.

- In The **Processor Setting**, change in the **Setting** field from DocumentType: IdRom (the last entry) to one of the types available in the list: AvailableDocumentTypes above DocumentType.
- Change the mapping as well, with the data received after calling the OCR service.

Example

This is applicable to Example Flow Setting.

Client script on edit mode form

To start the OCR component, use the loadComponent function from the FTOS.DFP library: loadComponent (componentName, processorSettings, recordId, existingFile).

```
var dfpHelper = ebs.importClientScript
('FTOS.DFP');
var componentName = 'FTOS_DFP_OCR'; //name of the
component

var recordId = formData.id;
```

```

var flowSettingsName = formData.FlowSettings;

var p = {};
var accountApplicationId = recordId;
p.accountApplicationId = accountApplicationId;
p.toStatus = "OCR in Progress";
ebs.callActionByName("FTOS_BARET_
AccountApplication_BusinessWorkflow", p,
function(e){

    var params = {};
    params.flowSettingsName = flowSettingsName;
    params.processorSettingsType = 'OCR';
    params.processorSettingsName = 'OCR_
CurrentAccount';

    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params,

function(f)
{
    var processorSettingsId =
f.UIResult.Data.ProcessorSettingsId;
    dfpHelper.loadComponent(componentName,
processorSettingsId, recordId, false);
});

});

```

Client script on insert mode form

To start the OCR component, use the loadComponent function from the FTOS.DFP library: loadComponent (componentName, processorSettings, recordId, existingFile). The recordId parameter is null, as we are running the component from a form in insert mode.

```

var dfpHelper = ebs.importClientScript
('FTOS.DFP');

```

```

var componentName = 'FTOS_DFP_OCR'; //name of the
component

var flowSettingsName = formData.FlowSettings;

var p = {};
var accountApplicationId = recordId;
p.accountApplicationId = accountApplicationId;
p.toStatus = "OCR in Progress";
ebs.callActionByName("FTOS_BARET_
AccountApplication_BusinessWorkflow", p,
function(e){
    var params = {};
    params.flowSettingsName = flowSettingsName;
    params.processorSettingsType = 'OCR';
    params.processorSettingsName = 'OCR_
CurrentAccount';

    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params,
function(f)
{
    var processorSettingsId =
f.UIResult.Data.ProcessorSettingsId;
    dfpHelper.loadComponent(componentName,
processorSettingsId, null, false);
    });
});

```

After the OCR process ends, the results are returned in sessionStorage in an item called ocrResult. After reading the results, we recommend to remove the object. It will be also removed when a new OCR component is instantiated.

```

var ocrResult = sessionStorage.getItem
("ocrResult");
ocrResult = JSON.parse(ocrResult);

if (ocrResult) {

```

```
ebs.setFormAttributeValue
("ebsContainerContent", "LastName",
ocrResult.updateObject.LastName);
ebs.setFormAttributeValue
("ebsContainerContent", "FirstName",
ocrResult.updateObject.FirstName);
sessionStorage.removeItem("ocrResult");
};
```

NOTE The Result of OCR execution will be saved in this entity: FTOS_DFP_OCR.

5. Face Recognition Configuration

For configuring Face Recognition, select **Digital Flow Processing > Flow Settings**. You have 1 flow as an example for Face Recognition.

Example

Javascript code - After Events

```
var componentName = "FTOS_DFP_
FaceRecognitionLiveness"; //name of the component
var recordId = ebs.getCurrentEntityId();
var fileExists = true; // "pictureOcr" argument =
source entity file attribute
var dfpHelper = ebs.importClientScript
('FTOS.DFP');
var flowSettingsName = formData.FlowSettings;

var p = {};
p.accountApplicationId = recordId;
p.toStatus = "Face Recognition in Progress";

ebs.callActionByName("FTOS_BARET_
AccountApplication_BusinessWorkflow", p,
function(e){
```

```

var params = {};
params.flowSettingsName = flowSettingsName;
params.processorSettingsType =
'FaceRecognition';
params.processorSettingsName =
'FaceRecognitionLiveness_Example';

ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params,
function(f)
{
var processorSettingsId =
f.UIResult.Data.ProcessorSettingsId;
console.log("processorSettingsName "
+ processorSettingsId);
dfpHelper.loadComponent
(componentName, processorSettingsId, recordId,
fileExists);
});
});

```

NOTE The Result of Face Recognition execution will be saved in this entity: FTOS_DFP_FaceRecognition.

6. Video Configuration

NOTE Make sure you have imported the second pack as well: Cognitive Processor Operator pack.

For the configuration of Video Configuration, select **Digital Flow Processing -> Flow Settings** . You have 1 flow as an example for Video.

NOTE You can copy the examples and modify them.

3 Upgrade Application Configuration

1. Import the Packs Again

Go to **DevOps > Deployment Packages**, and import the received packs. The **01_FTOS_DFP_OCR_FR** pack contains the services configuration model and scripts for OCR, Face Recognition.

2. Modify the Data

Import the data from **FTOS_DFP_OCR_FR .xlsx** files.

- Import_01_FTOS_DFP_FlowSettings:

- Import_01_FTOS_DFP_FlowSettings.xlsx

- Import_FTOS_DFP_ProcessorSettings:

- Import_02_FTOS_DFP_ProcessorSettings.xlsx

Using the example, please adapt your processors settings with the new structure found in the new examples.

NOTE For the OCR, Face Recognition, and Video components to work, the sites must run on HTTPS.

Setting Up a Face Recognition Automation Processor

IMPORTANT!

For the Face Recognition component to work, the sites must run on HTTPS.

1 Create a digital flow processor settings

The Face Recognition automation processor must be hosted inside a generic processor settings group. A generic processor settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a generic processor settings group you wish to host your Face Recognition automation processor, skip to "[2 Edit the Face Recognition automation processor](#)" below. Otherwise, follow the instructions below to create a new generic processor settings group:

1. In FintechOS Portal, click the main menu icon at the top left corner.
2. In the main menu, select **Digital Flow processing**.
3. Click **Flow settings**
4. In the Flow settings List page, click the **Insert** button at the top right corner to add a new flow settings.
5. In the Add Generic Processor Settings window, enter a **Name** for your generic processor settings.
6. Click the **Save and Close** button at the top right corner to save your flow settings.

2 Edit the Face Recognition automation processor

1. In the Flow settings List, double click the name of the setting created initially.
2. In the **Add Processor Settings** screen, fill in the following fields:
 - **Name** – Enter a name for your automation processor
 - **Digital Processor Type** – Select **Face Recognition**.
 - **Settings** – JSON code for the automation processor's settings. For details, see "[Face Recognition Settings](#)" on the next page.
 - **Mapping** – JSON code for the automation processor's mappings. For details, see "[Face Recognition Mappings](#)" on page 180.
3. Click the **Save and Close** button at the top right corner to save your automation processor.

Face Recognition Settings

The Face Recognition settings are defined in JSON format as key-value pairs. The following settings are available:

Setting	JSON Key	Description
Entity Name	SourceEntityName	The name of the entity where the Face Recognition process. Required only if the Face Recognition process starts from an edit form, to update the business status after the Face Recognition process ends.
	DestinationEntityName	Name of the entity that is populated with the data returned by face recognition.
Name	SourceLookupDestinationName	Name of the SourceEntityName lookup key that points to DestinationEntityName. If they are the same entity, enter the primary key.
File	FileAttributeName	The name of the file attribute where the first picture that will be used in the Compare Process, is saved (the image from the OCR process).
Maximum number of face recognition attempts	MaxRetry	The maximum number of face recognition attempts. If this number is reached, the user will be redirected according to the specifications in the " maskNextStepUrlFail " on page 179.
Confidence	MinimumAcceptedConfidence	The minimum confidence value that the Face Recognition process has to generate in order to consider the compare result a success. Value is a number between 0.00 and 1.00.

Setting	JSON Key	Description
Redirect in case of success	maskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful face recognition process.</p> <ul style="list-style-type: none"> • entity – Entity name. • form – Form name of the above entity. • section – Optional parameter for the section name of the above form. <p>The section can be specified using the name of the section or the section’s number. Example "section": "Personal Data" and "section": "1" are both valid.</p>

Setting	JSON Key	Description
Redirect in case of failure	maskNextStepUrlFail	<p>Location in the user interface where the user is redirected after the maximum number of failed face recognition attempts is reached (see "MaxRetry" on page 177).</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form. <p>The section can be specified using the name of the section or the section’s number. Example "section": "Current Account" and "section": "4" are both valid.</p>
Business status update in case of success	businessStatusSuccess	Business workflow status update of the "Entity Name" on page 177 if the face recognition process is successful.
Business status update in case of failure	businessStatusFail	Business workflow status update of the "Entity Name" on page 177 if the face recognition process fails.

NOTE
 For Liveness, please see ["Liveness" on page 181](#). In the Settings of the Face Recognition, it is possible to insert a key "isLiveness" : true to configure the feature.

Examples

Face Recognition settings

```

{
  "isLiveness": false,
  "DestinationEntityName": "Test1",
  "SourceEntityName": "Test",
  "SourceLookupDestinationName": "Test1id",
  "FileAttributeName": "pictureOcr",
  "MaxRetry": 5,
  "MinimumAcceptedConfidence": 0.2,
  "maskNextStepURLSuccess": {"entity": "Test", "form":
"FaceResponse" },
  "maskNextStepURLFail": {"entity": "Test", "form":
"FaceResponse" },
  "businessStatusSuccess": "Face Recognition Valid",
  "businessStatusFail": "Face Recognition Failed"
}
    
```

NOTE The Result of Face Recognition execution will be saved in the entity: **FTOS_DFP_FaceRecognition**.

Face Recognition Mappings

In the Mapping text-type field, specify the mapping between the attributes obtained by the Face Recognition component and the sourceEntity attributes that call the component.

Setting Name	Description
Map	Key-value pairs that match the field name as returned by the Face Recognition processor (the key) to the attribute name in the destination entity (the value).

Examples

Sample JSON code for Face Recognition mapping

```

{
  "Confidence": "confidence"
}
    
```

Liveness

It is a unique feature part of Face Recognition that offers effective access security by scanning the face of the user. The process is unique by detecting the face of a human being with the use of a camera either smart-phone, webcam or any incorporated camera without the need for sensors or additional peculiarities. It is most useful in the identification of a real individual. With such a unique way to determine real faces from photographs, Liveness can be used in multiple journeys when the identification of a client is mandatory for approval. It aids the institution to avoid fraud, and process the clients in an efficient way. Customer identification is the first step in any process before opening a contract.

How the process works

The detection is done by a complex process with four dimensions, X & Y and time to capture multiple 2D frames over a specific time spam. Lastly, with the use of AI to recreate a 3D representation of the person. The process involves capturing the 30-90 frontal frames to render a 3D object using a camera. It has a Certified Liveness Check (ISO 30107-3, DEA EPCS) as well.

Liveness feature is not a selfie, but a video that takes pictures of a human face from different angles to ensure an exact build of a 3D FaceMap. The algorithms pay attention to human features and how they are presented and ask the person to move closer to the camera to take further photos to confirm the presence of a human being.

The liveness detection is not fooled by:

- photos/ videos
- dolls
- masks
- people who look alike
- projections.

How to call constructor Liveness Component

In a custom Digital Journey, please insert the following JavaScript:

```

const livenessConfig: ILivenessConfig = {
  containerElement: document.getElementById("livenessContainer")
  resourcesLocation: "",
  localization: {},
  uiCustomization: {},
  globalState: {},
  afterInitCallback: data => {},
  maxCounter: 3,
  requests: {
    getSessionToken: (userAgentString) => {
      return
      ebs.callActionByNameAsync("FTOS_DCS_Liveness_
SessionToken", {
        userAgentString: userAgentString
      })
    },
    getServerResponse: (userAgentString, parameters) => {
      return
      ebs.callActionByNameAsync("FTOS_DCS_Liveness_Check", {
        userAgentString: userAgentString,
        parameters: parameters
      })
    },
    cancelInFlightRequests: (() => {});
  };
};
const liveness = new LivenessComponent(livenessConfig);

```

How to call the Liveness processor within the Face Recognition processor

In the processor settings, in the settings code editor, insert the key `isLiveness : true` to enable the feature.

How to change the text and colour

The Liveness feature makes it possible to edit the text and colour of the UI a client will be interacting with.

To do so, access the client library `Ftos_dfp_liveness` and change the two UI elements using the following JavaScript:

```

interface ILivenessUi {

```

```

    // Customize the ZoOm Oval and the ZoOm Progress Spinner
    animations.
    ovalCustomization: ZoomOvalCustomization;
    // Customize the ZoOm Feedback Bar.
    feedbackCustomization: ZoomFeedbackBarCustomization;
    // Customize the ZoOm Frame.
    frameCustomization: ZoomFrameCustomization;
    // Customize the ZoOm Frame exit animation.
    exitAnimationCustomization: ZoomExitAnimationCustomization;
    // Customize the ZoOm Cancel Button.
    cancelButtonCustomization: ZoomCancelButtonCustomization;
    // Customize the time after which the ZoOm Session should
    timeout.
    sessionTimerCustomization: ZoomSessionTimerCustomization;
    // Customize the Loading Spinner and the text shown to the user
    while the camera loads.
    initialLoadingAnimationCustomization:
ZoomInitialLoadingAnimationCustomization;
    // Customize the New User Guidance and Retry Screens.
    guidanceCustomization: ZoomGuidanceCustomization;
    // Customize the ZoOm Overlay, separating the ZoOm Interface
    from the presenting application context.
    overlayCustomization: ZoomOverlayCustomization;
    // Customize the Result Screen.
    resultScreenCustomization: ZoomResultScreenCustomization;
    // Customize the ZoOm Identity Check Screens.
    idScanCustomization: ZoomIDScanCustomization;
    // Show Camera Permissions Denied Screen.
    enableCameraPermissionsHelpScreen: boolean;
    // Force the oval stroke to be drawn as opaque.
    shouldDrawOvalStrokeOpaque: boolean;
    // This function allows special runtime control of the success
    message shown when the success animation occurs. Please note that
    you can also customize this string via the standard
    customization/localization methods provided by ZoOm. Special
    runtime access is enabled to this text because the developer may
    wish to change this text depending on ZoOm's mode of
    operation. Default is in the customizable localization string "zoom_
    result_success_message"
    static setOverrideResultScreenSuccessMessage: (message: string)
=> void;
}
interface ZoomOvalCustomization {
    // Color of the ZoOm Oval outline. Default is white.
    strokeColor: string;
    // Color of the animated ZoOm Progress Spinner strokes. Default
    is custom ZoOm color.

```

```

    progressColor1: string;
    progressColor2: string;
    // Thickness of the animated ZoOm Progress Spinner strokes.
    Default is dynamically configured per device at runti
    progressStrokeWidth: number;
    // Thickness of the ZoOm Oval outline. Default is dynamically
    configured per device at runtime.
    strokeWidth: number;
}
interface ZoomFeedbackBarCustomization {
    // Color of the ZoOm Feedback Bar's background. Recommend
    making this have some transparency. Default is custom ZoOm color.
    backgroundColor: string;
    // Color of the text displayed within the ZoOm Feedback Bar.
    Default is white.
    textColor: string;
    // Font of the text displayed within the ZoOm Feedback Bar.
    textFont: string;
    // Spacing between the characters of the text displayed within
    the ZoOm Feedback Bar. Accepts any value assignable to the
    LetterSpacing CSS attribute. Default is 'normal'.
    textSpacing: string;
    // Corner radius of the ZoOm Feedback Bar. Default is
    dynamically configured per device at runtime.
    cornerRadius: string;
    // Shadow displayed behind the ZoOm Feedback Bar. This accepts
    box-shadow css attribute string values. Default is a custom sized
    black shadow.
    shadow: string;
    // Control whether to enable the pulsating-text animation
    within the ZoOm Feedback Bar. Default is true (enabled).
    enablePulsatingText: boolean;
    // Control the percent of the available ZoOm Frame width to use
    for the ZoOm Feedback Bar's width on desktop browsers. Relative
    width percent is represented in decimal notation, ranging from 0.0
    to 1.0. If the value configured is equal to or greater than 1.0,
    the ZoOm Feedback Bar will be drawn to at max width within the ZoOm
    Frame. If the value configured results in a width that is less than
    the minimum width, which is 2x the ZoOm Feedback Bar's height, then
    the ZoOm Feedback Bar's width will be set at the minimum. Default
    is dynamically configured per device at runtime.
    relativeWidthOnDesktop: string;

```



```

    // Control the percent of the available ZoOm Frame width to use
    // for the ZoOm Feedback Bar's width on mobile browsers. Relative
    // width percent is represented in decimal notation, ranging from 0.0
    // to 1.0. If the value configured is equal to or greater than 1.0,
    // the ZoOm Feedback Bar will be drawn to at max width within the ZoOm
    // Frame. If the value configured results in a width that is less than
    // the minimum width, which is 2x the ZoOm Feedback Bar's height, then
    // the ZoOm Feedback Bar's width will be set at the minimum. Default
    // is dynamically configured per device at runtime.

```

```

    relativeWidth: string;
}
interface ZoomFrameCustomization {
    // Shadow displayed behind the ZoOm Frame. This accepts box
    // shadow css attribute string values. Default is none.
    shadow: string;
    // Color of the ZoOm Frame's border. Default is white.
    borderColor: string;
    // Corner radius of the ZoOm Frame. Default is dynamically
    // configured per device at runtime.
    borderCornerRadius: string;
    // Thickness of the ZoOm Frame's border. Default is dynamically
    // configured per device at runtime.
    borderWidth: string;
    // Color of the background surrounding the oval outline during
    // ZoOm. Default is custom ZoOm color.
    backgroundColor: string;
    // Applies a blur effect over the background surrounding the
    // oval outline during ZoOm. Default is off.
    blurEffectStyle: string;
}
interface ZoomExitAnimationCustomization {
    // Customize the transition out animation for a successful ZoOm
    // Session. */
    exitAnimationSuccess: 0 | 1 | 2; // None = 0, RippleOut =
    // 1, FadeOutMin = 2
    // Customize the transition out animation for an unsuccessful
    // ZoOm Session. */
    exitAnimationUnsuccess: 0 | 1 | 2; // None = 0, RippleOut =
    // 1, FadeOutMin = 2
}
interface ZoomCancelButtonCustomization {
    // Location, or use, of the ZoOm Cancel Button. Default is
    // ZoomCancelButtonLocation.TopLeft.
    location: ZoomCancelButtonLocation;
}

```

```

    // The size and location of the cancel button within the
    current screen's bounds. Configure using the convenience method
    .setCustomLocation(x:number, y:number, width:number,
    height:number). Note: In order to use a custom-located cancel
    button, you MUST set .location to the enum value
    ZoomCancelButtonLocation.Custom. Default is set at origin 0,0 with
    a size of 0 by 0.
    customLocation: ZoomRect;
    // Image displayed on the ZoOm Cancel Button. Default is
    configured to use image named 'zoom_cancel' located in
    '/zoomimages/' directory (or custom configured default directory for
    ZoOm images).
    customImage: string;
}
interface ZoomSessionTimerCustomization {
    maxTimeOverall: number;
    maxTimeToDetectFirstFace: number;
    maxTimeToDetectFirstFaceInPhaseTwo: number;
    maxTimeBeforeCameraPermissionsError: number;
}
interface ZoomInitialLoadingAnimationCustomization {
    // HTMLElement displayed while camera is loading. Default is a
    custom animated loading spinner and text.
    element: HTMLElement;
}
interface ZoomGuidanceCustomization {
    // Thickness of the action button's border during the New User
    Guidance and Retry Screens. Default is dynamically configured per
    device at runtime.
    buttonBorderWidth: string;
    // Color of the action button's border during the New User
    Guidance and Retry Screens. Default is transparent.
    buttonBorderColor: string;
    // Corner radius of the action button's border during the New
    User Guidance and Retry Screens. Default is dynamically configured
    per device at runtime.
    buttonCornerRadius: string;
    // Color of the action button's text during the New User
    Guidance and Retry Screens. Default is white.
    buttonTextNormalColor: string;
    // Color of the action button's text when the button is pressed
    during the New User Guidance and Retry Screens. Default is white.
    buttonTextHighlightColor: string;
    // Color of the action button's text when the button is
    disabled during the New User Guidance and Retry Screens. Default is
    white.
    buttonTextDisabledColor: string;
}

```

```

    // Color of the action button's background during the New User
    Guidance and Retry Screens. Default is custom ZoOm color.
    buttonBackgroundNormalColor: string;
    // Color of the action button's background when the button is
    pressed during the New User Guidance and Retry Screens. Default is
    custom ZoOm color.
    buttonBackgroundHighlightColor: string;
    // Color of the action button's background when the button is
    disabled during the New User Guidance and Retry Screens. Default is
    custom ZoOm color.
    buttonBackgroundDisabledColor: string;
    // Font of the title's text during the New User Guidance and
    Retry Screens.
    headerFont: string;
    // Spacing between the characters of the title's text during
    the New User Guidance and Retry Screens. Accepts any value
    assignable to the LetterSpacing CSS attribute. Default is 'normal'.
    headerTextSpacing: string;
    // Font of the title's subtext and messages during the New User
    Guidance and Retry Screens.
    subtextFont: string;
    // Spacing between the characters of the title's subtext and
    messages during the New User Guidance and Retry Screens. Accepts
    any value assignable to the LetterSpacing CSS attribute. Default is
    'normal'.
    subtextTextSpacing: string;
    // Font of the title's subtext during the New User Guidance and
    Retry Screens. Default is a bold system font.
    buttonFont: string;
    // Spacing between the characters of the action button's text
    during the New User Guidance and Retry Screens. Accepts any value
    assignable to the LetterSpacing CSS attribute. Default is 'normal'.
    buttonTextSpacing: string;
    // Control the percent of the available ZoOm Frame width to use
    for the action button during the New User Guidance and Retry
    Screens for mobile browsers. Relative width percent is represented
    in decimal notation, ranging from 0.0 to 1.0. If the value
    configured is equal to or greater than 1.0, the action button will
    be drawn to at max width within the ZoOm Frame. If the value
    configured results in a width that is less than the action button's
    height, the action button's width will equal its height. Default is
    dynamically configured per device at runtime.
    buttonRelativeWidth: string;

```

```

    // Control the percent of the available ZoOm Frame width to use
    for the action button during the New User Guidance and Retry
    Screens for desktop browsers. Relative width percent is represented
    in decimal notation, ranging from 0.0 to 1.0. If the value
    configured is equal to or greater than 1.0, the action button will
    be drawn to at max width within the ZoOm Frame.If the value
    configured results in a width that is less than the action button's
    height, the action button's width will equal its height. Default is
    dynamically configured per device at runtime.
    buttonRelativeWidthOnDesktop: string;
    // Color of the background for the New User Guidance and Retry
    Screens. Default is white.
    backgroundColors: string;
    // Color of the text displayed on the New User Guidance and
    Retry Screens (not including the action button text). Default is
    custom ZoOm color.
    foregroundColor: string;
    // Color of the Get Ready To ZoOm Screen's oval fill. Default
    is transparent.
    readyScreenOvalFillColor: string;
    // Background color of the Get Ready To ZoOm Screen text views
    during the New User Guidance and Retry Screens. This will only be
    visible when text is detected as overlapping or too close with the
    Ready screen oval. Default is a semi-opaque shade of black.
    readyScreenTextBackgroundColor: string;
    // Background corner radius of the Get Ready To ZoOm Screen
    text views during the New User Guidance and Retry Screens. This
    will only be visible when text is detected as overlapping or too
    close with the Get Ready To ZoOm Screen's oval. Default is
    dynamically configured per device at runtime.
    readyScreenTextBackgroundCornerRadius: string;
    // Image displayed as Ideal ZoOm example (right image) during
    the first Retry Screen. Default is configured to use image named
    'zoom_ideal' located in '/zoom-images/' directory (or custom
    configured default directory for ZoOm images).
    retryScreenIdealZoomImage: string;
    // Images displayed as Ideal ZoOm examples (right image) during
    the first Retry Screen. When configured to a non-empty array, these
    images will override the single image configured for
    imageCustomization.idealZoomImage. Default is an empty array.
    retryScreenSlideshowImages: string[];
    // Control the time that each image is shown for before
    transitioning to the next image. Default is 1500ms.
    retryScreenSlideshowInterval: string;
    // Control whether to allow the slideshow images to appear in a
    randomized order during each Retry Screen. Default is true
    (enabled).
    enableRetryScreenSlideshowShuffle: boolean;

```

```

    // Color of the image borders during the first Retry Screen.
    Default is custom ZoOm color.
    retryScreenImageBorderColor: string;
    // Thickness of the image borders during the first Retry
    Screen. Default is dynamically configured per device at runtime.
    retryScreenImageBorderWidth: string;
    // Corner radius of the image borders during the first Retry
    Screen. Default is dynamically configured per device at runtime.
    retryScreenImageCornerRadius: string;
    // Color of the oval's stroke that overlay's the Ideal image
    example during the first Retry Screen. Default is white.
    retryScreenOvalStrokeColor: string;
    // Control whether to layout the Retry Screen's instruction
    messages using bullet-points. Applicable localized instruction
    message strings include: zoom_retry_instruction_message_1, zoom_
    retry_instruction_message_2, zoom_retry_instruction_message_3. If
    enabled, each instruction message will be placed on a new line,
    preceded with a bullet-point, and will not extend to multiple
    lines. If disabled, all instruction messages will be concatenated
    into a multi-line string. Default is true (enabled).
    enableRetryScreenBulletedInstructions: boolean;
    // Image displayed on the Camera Permissions Screen. Default is
    configured to use image named 'zoom_camera' located in
    '/zoomimages/' directory (or custom configured default directory for
    ZoOm images).
    cameraPermissionsScreenImage: string;
}
interface ZoomOverlayCustomization {
    // Color of the ZoOm Overlay background. Default is white.
    backgroundColor: string;
    // Color of the text shown on ZoOm Overlay. This includes the
    Low Light Mode Toggle's text color, which only applies to desktop
    browsers. Default is custom ZoOm color.
    foregroundColor: string;
    // Applies a blur effect over the background of the ZoOm
    Overlay. Default is off.
    blurEffectStyle: string;
    // Control whether to show the branding image the ZoOm Frame on
    top of the ZoOm Overlay.<br> Default is true (shown).
    showBrandingImage: boolean;
    // Image displayed below the ZoOm Frame on top of the ZoOm
    Overlay. Default is configured to use image named 'zoom_your_app_
    Logo' located in '/zoom-images/' directory (or custom configured
    default directory for ZoOm images).
    brandingImage: string;
}
interface ZoomResultScreenCustomization {
    // Color of the Result Screen's background. Default is white.

```

```

    backgroundColors: string;
    // Color of the text displayed on the Result Screen. Default is
    custom ZoOm color.
    foregroundColor: string;
    // Color of the result animation's background. Default is
    custom ZoOm color.
    resultAnimationBackgroundColor: string;
    // Color of the result animation's accent color. Default is
    white.
    resultAnimationForegroundColor: string;
    // Image displayed behind the result foreground animation for
    success scenarios. If image is configured, default result
    background animation will be hidden. Default is set to an empty
    string and will fallback to using the default result background
    animation, which respects the color assigned to
    .resultAnimationBackgroundColor.
    resultAnimationSuccessBackgroundImage: string;
    // Image displayed behind the result foreground animation for
    unsuccess scenarios. If image is configured, default result
    background animation will be hidden. Default is set to an empty
    string and will fallback to using the default result background
    animation, which respects the color assigned to
    .resultAnimationBackgroundColor.
    resultAnimationUnsuccessBackgroundImage: string;
    // Font of the message text displayed on the Result Screen.
    messageFont: string;
    // Spacing between the characters of the message text displayed
    on the Result Screen. Accepts any value assignable to the
    LetterSpacing CSS attribute. Default is 'normal'.
    messageTextSpacing: string;
    // Color of the activity indicator animation shown during
    server-side work. Default is custom ZoOm color.
    activityIndicatorColor: string;
    // Image displayed and rotated during server-side work. If
    image is configured, default activity indicator will be hidden.
    Default is set to an empty string and will fallback to using
    default activity indicator animation.
    customActivityIndicatorImage: string;
    // Control the speed of the rotation for your custom activity
    indicator image. Only applicable when image is configured for
    .customActivityIndicatorImage. This value indicates the duration of
    each full rotation. Default is 1s.
    customActivityIndicatorRotationInterval: string;
    // Control whether to show or hide the upload progress bar
    during server-side work. Default is true (shown).
    showUploadProgressBar: boolean;
    // Color of the upload progress bar's fill. Default is custom
    ZoOm color.
    uploadProgressFillColor: string;

```

```

    // Color of upload progress bar's track. Default is a semi
    opaque shade of black.
    uploadProgressTrackColor: string;
}
interface ZoomIDScanCustomization {
    // Image displayed on the ID Scan Select ID Document page
    Default is configured to use image named 'zoom_branding_logo_id_
    check' located in '/zoom-images/' directory (or custom configured
    default directory for Zoom images).
    showSelectionScreenBrandingImage: boolean;
    // Color of the text displayed on the Identity Document Type
    Selection Screen (not including the action button text). Default is
    off-black.
    selectionScreenForegroundColor: string;
    // Font of the title during the Identity Document Type
    Selection Screen.
    headerFont: string;
    // Spacing between the characters of the title's text during
    the Identity Document Type Selection Screen. Accepts any value
    assignable to the LetterSpacing CSS attribute. Default is 'normal'.
    headerTextSpacing: string;
    // Font of the message text during the Identity Document
    Capture and Review Screens.
    subtextFont: string;
    // Spacing between the characters of the title's subtext and
    messages during the Identity Document Capture and Review Screens.
    Accepts any value assignable to the LetterSpacing CSS attribute.
    Default is 'normal'.
    subtextTextSpacing: string;
    // Font of the action button's text during the Identity Check
    Screens.
    buttonFont: string;
    // Spacing between the characters of the action button's text
    during the Identity Check Screens. Accepts any value assignable to
    the LetterSpacing CSS attribute. Default is 'normal'.
    buttonTextSpacing: string;
    // Thickness of the action button's border during the Identity
    Check Screens Default is dynamically configured per device at
    runtime.
    buttonBorderWidth: string;
    // Color of the action button's border during the Identity
    Check Screens Default is transparent.
    buttonBorderColor: string;
    // Corner radius of the action button's border during the
    Identity Check Screens. Default is dynamically configured per
    device at runtime.
    buttonCornerRadius: string;

```

```

    // Color of the action button's text during the Identity Check
    Screens. Default is white.
    buttonTextNormalColor: string;
    // Color of the action button's text when the button is pressed
    during the Identity Check Screens. Default is white.
    buttonTextHighlightColor: string;
    // Color of the action button's text when the button is
    disabled during the Identity Check Screens. Default is white.
    buttonTextDisabledColor: string;
    // Color of the action button's background during the Identity
    Check Screens. Default is custom ZoOm color.
    buttonBackgroundNormalColor: string;
    // Color of the action button's background when the button is
    pressed during the Identity Check Screens. Default is custom ZoOm
    color.
    buttonBackgroundHighlightColor: string;
    // Color of the action button's background when the button is
    disabled during the Identity Check Screens. Default is custom ZoOm
    color.
    buttonBackgroundDisabledColor: string;
    // Control the percent of the available ZoOm Frame width to use
    for the action button during the Identity Check Screens for mobile
    browsers. Relative width percent is represented in decimal
    notation, ranging from 0.0 to 1.0. If the value configured is equal
    to or greater than 1.0, the action button will be drawn to at max
    width within the ZoOm Frame. If the value configured results in a
    width that is less than the action button's height, the action
    button's width will equal its height. Note: The Identity Document
    Review Screen action buttons will be drawn at half the configured
    width. Default is dynamically configured per device at runtime.
    buttonRelativeWidth: string;
    // Control the percent of the available ZoOm Frame width to use
    for the action button during the Identity Check Screens for desktop
    browsers. Relative width percent is represented in decimal
    notation, ranging from 0.0 to 1.0. If the value configured is equal
    to or greater than 1.0, the action button will be drawn to at max
    width within the ZoOm Frame. If the value configured results in a
    width that is less than the action button's height, the action
    button's width will equal its height. Note: The Identity Document
    Review Screen action buttons will be drawn at half the configured
    width. Default is dynamically configured per device at runtime.
    buttonRelativeWidthOnDesktop: string;
    // Color of the Identity Document Type Selection Screen
    background. Default is white.
    selectionScreenBackgroundColors: string;
    // Applies a blur effect over the background of the Identity
    Document Type Selection Screen. Default is off.
    selectionScreenBlurEffectStyle: string;

```



```

    // Image displayed on the Identity Document Type Selection
    Screen. Default is configured to use image named 'zoom_branding_
    Logo_id_check' located in '/zoom-images/' directory (or custom
    configured default directory for ZoOm images).
    selectionScreenBrandingImage: string;
    // Color of the text displayed on the Identity Document Capture
    Screen (not including the action button text). Default is white.
    captureScreenForegroundColor: string;
    // Color of the text view background during the Identity
    Document Capture Screen. Default is custom ZoOm color.
    captureScreenTextBackgroundColor: string;
    // Color of the text view background border during the Identity
    Document Capture Screen. Default is transparent.
    captureScreenTextBackgroundBorderColor: string;
    // Thickness of the text view background border during the
    Identity Document Capture Screen. Default is 0.
    captureScreenTextBackgroundBorderWidth: string;
    // Corner radius of the text view background and border during
    the Identity Document Capture Screen. Default is dynamically
    configured per device at runtime.
    captureScreenTextBackgroundCornerRadius: string;
    // Color of the Identity Document Capture Screen's background.
    Default is white.
    captureScreenBackgroundColor: string;
    // Color of the Identity Document Capture Frame's stroke.
    Default is custom ZoOm color.
    captureFrameStrokeColor: string;
    // Thickness of the Identity Document Capture Frame's stroke.
    Default is dynamically configured per device at runtime.
    captureFrameStrokeWidth: string;
    // Corner radius of the Identity Document Capture Frame.
    Default is dynamically configured per device at runtime.
    captureFrameCornerRadius: string;
    // Color of the text displayed on the Identity Document Review
    Screen (not including the action button text). Default is white.
    reviewScreenForegroundColor: string;
    // Color of the text view background during the Identity
    Document Review Screen. Default is custom ZoOm color.
    reviewScreenTextBackgroundColor: string;
    // Color of the text view background border during the Identity
    Document Review Screen. Default is transparent.
    reviewScreenTextBackgroundBorderColor: string;
    // Thickness of the text view background border during the
    Identity Document Review Screen. Default is 0.
    reviewScreenTextBackgroundBorderWidth: string;
    // Corner radius of the text view background and border during
    the Identity Document Review Screen. Default is dynamically
    configured per device at runtime.

```

```

reviewScreenTextBackgroundBorderCornerRadius: string;
// Corner radius of the ID Document Preview image displayed on
the Identity Document Review Screen. Default is dynamically
configured per device at runtime.
reviewScreenDocumentPreviewCornerRadius: string;
// Color of the Identity Document Review Screen background.
Default is white.
reviewScreenBackgroundColors: string;
// Applies a blur effect over the background of the Identity
Document Review Screen. Default is off.
reviewScreenBlurEffectStyle: string;
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is an ID Card. This image
acts as a placeholder to show a status of incomplete for capturing
the ID Card's front side. This only applies to desktop browsers.
Default is configured to use image named 'zoom_id_card_placeholder_
front' located in '/zoom-images/' directory (or custom configured
default directory for ZoOm images).
captureScreenIDFrontPlaceholderImage: string;
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is an ID Card. This image
acts as a placeholder to show a status of incomplete for capturing
the ID Card's back side. This only applies to desktop browsers.
Default is configured to use image named 'zoom_id_card_placeholder_
back' located in '/zoom-images/' directory (or custom configured
default directory for ZoOm images).
captureScreenIDBackPlaceholderImage: string;
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is a Passport. This image
acts as a placeholder to show a status of incomplete for capturing
the Passport. This only applies to desktop browsers. Default is
configured to use image named 'zoom_passport_placeholder' located
in '/zoom-images/' directory (or custom configured default directory
for ZoOm images).
captureScreenPassportPlaceholderImage: string;
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is an ID Card. This image
acts as a placeholder to show a status of complete for capturing
the ID Card's front side. This only applies to desktop browsers.
Default is configured to use image named 'zoom_id_front_checkmark'
located in '/zoom-images/' directory (or custom configured default
directory for ZoOm images).
captureScreenIDFrontCheckmarkImage: string;

```

```
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is an ID Card. This image
acts as a placeholder to show a status of complete for capturing
the ID Card's back side. This only applies to desktop browsers.
Default is configured to use image named 'zoom_id_back_checkmark'
located in '/zoom-images/' directory (or custom configured default
directory for ZoOm images).
```

```
captureScreenIDBackCheckmarkImage: string;
```

```
// Image displayed below the ZoOm Frame during Identity Check
when the Identity Document Type selected is a Passport. This image
acts as a placeholder to show a status of complete for capturing
the Passport. This only applies to desktop browsers. Default is
configured to use image named 'zoom_passport_checkmark' located in
'/zoom-images/' directory (or custom configured default directory
for ZoOm images).
```

```
captureScreenPassportCheckmarkImage: string;
```

```
}
```

Customization of Localization

It is possible to target the Liveness feature to a specific cultural group by translating the messages from a language to another, setting the dates in a particular manner and complying with local customs.

```
interface ILivenessLocalization {
    zoom_accessibility_cancel_button: string, // 'Anulare'
    zoom_feedback_center_face: string, // 'Centrati
fata'
    zoom_feedback_face_not_found: string, // 'Incadrati
fata in chenar',
    zoom_feedback_move_phone_away: string, //
'Indepartati-va',
    zoom_feedback_move_away_web: string, //
'Indepartati-va',
    zoom_feedback_move_phone_closer: string, //
'Apropiati-va',
    zoom_feedback_move_phone_to_eye_level: string, //
'Mutati telefonul la nivelul ochilor',
    zoom_feedback_move_to_eye_level_web: string, //
'Uitati-va direct in camera',
    zoom_feedback_face_not_looking_straight_ahead:
string, // 'Uitati-va drept in fata',
```

```

        zoom_feedback_face_not_upright: string, // 'Tineti
        capul drept',
        zoom_feedback_face_not_upright_mobile: string, //
        'Tineti capul drept',
        zoom_feedback_hold_steady: string, // 'Stati
        nemiscat',
        zoom_feedback_move_web_closer: string, //
        'Apropiati-va',
        zoom_feedback_move_web_even_closer: string, // 'Mai
        aproape',
        zoom_feedback_use_even_lighting: string, //
        'Illuminati fata uniform',
        zoom_instructions_header_ready: string, //
        'Pregatiti-va pentru selfi-ul video',
        zoom_instructions_message_ready: string, //
        'Incadrați-va fata in ovalul mic si dupa in ovalul mare',
        zoom_action_im_ready: string, // 'SUNT PREGATIT',
        zoom_result_facemap_upload_message: string, //
        'Incarcare<br/>Criptata<br/>3D FaceMap',
        zoom_result_idscan_upload_message: string, //
        'Incarcare<br/>Criptata<br/>ID Document',
        zoom_retry_header: string, // 'Hai sa mai incercam',
        zoom_retry_subheader_message: string, // 'Dar mai
        intai, uitati va la selfi-ul dumneavoastra si corectati
        mediul',
        zoom_retry_your_image_label: string, // 'Selfi-ul
        dumneavoastra',
        zoom_retry_ideal_image_label: string, // 'Pozitie
        ideala',
        zoom_retry_instruction_message_1: string, //
        'Expresie Neutra, Fara Zambete',
        zoom_retry_instruction_message_2: string, // 'Fara
        stralucire sau iluminare excesiva',
        zoom_retry_instruction_message_3: string, // '',
        zoom_action_ok: string, // 'OK',
        zoom_camera_permission_header: string, // 'Activati
        Camera',
        zoom_camera_permission_message: string, //
        'Permisuniile pentru camera sunt dezactivate. Va rugam sa
        verificati sistemuldumneavoastra de operare si setarile de
        browser. Pentru mai multe detalii accesati urmatorul link:',
        zoom_browser_camera_help_action_link: string, //
        'https://dev.zoomlogin.com/zoomsdk/#/browser-camera-help',
        zoom_camera_permission_launch_settings: string, //
        'Lanseaza Ajutorul de Camera',

```

```

        zoom_initializing_camera: string, // 'Initializare
Camera...',
        zoom_idscan_type_selection_header: string, //
'Selecteazar<br>Tipul Documentului',
        zoom_action_select_id_card: string, // 'CARTE
IDENTITATE',
        zoom_action_select_passport: string, // 'PASAPORT',
        zoom_idscan_capture_id_card_front_instruction_
message: string, // 'Aratati Fata Cartii de Identitate',
        zoom_idscan_capture_id_card_back_instruction_
message: string, // 'Aratati Spatele Cartii de Identitate',
        zoom_idscan_capture_passport_instruction_message:
string, // 'Aratati Pagina cu Poza din Pasaport',
        zoom_action_take_photo: string, // 'FOTOGRAFIAZA',
        zoom_idscan_review_id_card_front_instruction_
message: string, // 'Confirmati ca documentul este lizibil',
        zoom_idscan_review_id_card_back_instruction_message:
string, // 'Confirmati ca documentul este lizibil',
        zoom_idscan_review_passport_instruction_message:
string, // 'Confirmati ca documentul este lizibil',
        zoom_action_accept_photo: string, // 'ACCEPTARE',
        zoom_action_retake_photo: string, // 'RELUATI',
        zoom_result_idscan_unsuccess_message: string, //
'Poza din Document<br/>Nu se potriveste<br>Cu Selfi-ul',
        zoom_result_success_message: string, // 'Succes!',
    }

```

Adding Face Recognition to a Digital Journey

1. In Innovation Studio, create a custom journey to define a button to call the Face Recognition automation processor. For information on how to create a custom journey, see *FintechOS Studio User Guide*, page [Creating Custom Flows](#).
2. Go to the form driven journey or form step on which you want to add the button to call the Face Recognition automation processor.
3. Click the **Advanced** tab.

4. Click the **After Events** tab (for Steps, it is displayed by default, being the only tab available).
5. In the JavaScript field, provide the following code:

```

//import client library
var dfpHelper = ebs.importClientScript('FTOS.DFP');

//call FaceRecognition component custom form
var componentName = 'FTOS_DFP_FaceRecognition'; //name of the
component

var entitydata=ebs.getCurrentEntityData();

var recordId = formData.id;

var flowSettingsName = formData.FlowSettings;

var p = {};

var accountApplicationId = recordId;

p.accountApplicationId = accountApplicationId;

//p.toStatus = "Video Call in Progress";

//ebs.callActionByName("FTOS_Test_Processors_
BusinessWorkflow", p, function(e){

    var params = {};

    params.flowSettingsName = flowSettingsName;

    params.processorSettingsType = 'FaceRecognition';

    var typeOffacerecId =entitydata.FTOS_Test_Processors_
typeOfFaceRecid_displayname;

    switch (typeOffacerecId){

        case 'Face Recognition':

```

```

        params.processorSettingsName=
'FaceRecognition_FTOS_Test_Processors';

        break;

        case 'Liveness':

            params.processorSettingsName='Liveness';

            componentName='FTOS_DFP_
FaceRecognitionLiveness'

            break;

            case 'Liveness OCR':

                params.processorSettingsName='Liveness_OCR';

                componentName='FTOS_DFP_LivenessOCR'

                break;

        };

        ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", params, function(f)

        {

            var processorSettingsId =
f.UIResult.Data.ProcessorSettingsId;

            dfpHelper.loadComponent(componentName,
processorSettingsId, recordId, true);

        });

```

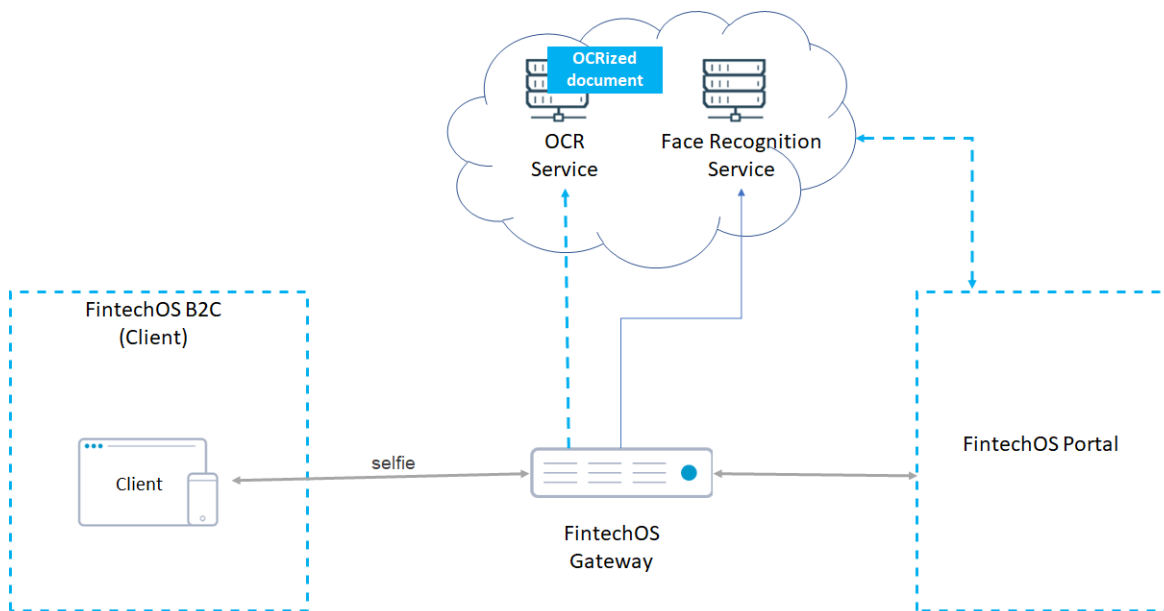
6. Click the **Save and Close** button at the top right corner to save your digital journey.

Video Streaming

Video Streaming is the continuous transmission of video content between a customer and a server. It facilitates business processes such as digital onboarding. The customer can start a live video call with a bank consultant/ call center operator using the FintechOS Video Identification embedded capability. The Video Streaming automation processor provides a seamless experience for customers and businesses enabling a predefined process to complete the customer identity verification or provide advice.

The video streaming flow:

1. The customer initiates a video streaming session.
2. The FintechOS Gateway tells the Video Streaming Service to create a video session.
3. The Video Streaming Service creates a session and sends the session ID back to the FintechOS Gateway.
4. The FintechOS Gateway creates an unique token and sends the session information to the FintechOS Portal and the customer is added into the Queue.
5. When an operator picks up the customer from the Queue, the FintechOS Gateway sends the session ID and token to the customer.
6. The customer uses the token to connect to the session.
7. The video session starts and the participants (customer and operator) can publish and subscribe to the video files being transmitted in real-time in the session.



Video Streaming Processor Features

- Embedded high quality and scalable video communications in the context of FintechOS web and mobile applications.
- Dynamically prioritize audio in response to network quality.
- Audio detection to control stream layout and display.
- Media streams in transit and at rest are encrypted using AES 128-bit encryption.

Applications

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes

- Claims handling
- Mortgage processing.

Installing Video Streaming

1 Install the SysPacks

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.
5. Download the **SySDigitalSolutionPackages v22.1.1000.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page,

2 Set up the Video Streaming Processor subscription key(s)

On the FintechOS Portal server, open the *web.config* file in a text editor and add the following entries in the <appSettings> section, depending on your subscription keys:

- Generic subscription key for Face Recognition Processor services:

```
<add key="FTOSServicesEndpoint" value="URL to the services endpoint"/>
<add key="FTOSServicesAppId" value="service authentication key"/>
```

- Automation Blocks Processor service subscription keys:

```
<add key="FTOSServicesVideoFaceEndpoint" value="URL to the services endpoint"/>
<add key="FTOSServicesVideoAppId" value="service authentication key"/>
```

Setting Up a Video Streaming Automation Processor

1 Add queues and operators

For every anonymous digital journey which require a video validation (onboarding, credit card loan, etc.), add a queue and operators who should handle the queue.

IMPORTANT!

For Video Streaming to work, the sites must run on HTTPS.

1. Log into the FintechOS Portal.
2. Click the main menu icon at the top left corner.
3. In the main menu, click **Digital Onboarding Configuration > Queues** . The Queues List page appears.
4. At the top-right corner of the page, click the **Insert** icon. The Add FTOS_DFP_QUEUE page appears.
5. Provide a **Name** for the queue and optionally a **Hello Message Text** to be displayed when customers initialize a video call with the operators.
6. At the top-right corner click the **Save and close** icon to save the queue.
7. Click the main menu icon at the top left corner.
8. In the main menu, click **Digital Onboarding Configuration > Operators**. The Operators List page appears.
9. At the top-right corner of the page, click the **Insert** icon. The ADD FTOS_DFP_OPERATOR page appears.
10. Type the operator **Name** and from the **Related User** field, select the FintechOS Portal user account of the person who will be fulfilling the operator role.

11. At the top right corner of the page, click the **Save and reload** icon. The EDIT FTOS_DFP_OPERATOR appears and the QUEUES section is unlocked.
12. In the QUEUES list, click the **Insert existing** button. A pop-up appears listing the existing queues.
13. In the pop-up, double click on the queue (you added following steps 4 to 6) . The selected queue is listed in the QUEUES section.
14. Click the **Save and Close** button () at the top right corner to save the settings.

The user who you selected to be operator for the queue will see the queue by clicking in the main menu **Digital Onboarding > My Queues**.

2 Create a digital flow processing settings group

The Video Streaming automation processor must be hosted in a flow settings group. A flow settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a flow settings group you wish to host your Video Streaming automation processor, skip to "[3 Add the Video Streaming automation processor to flow settings group](#)" on the next page. Otherwise, follow the instructions below to create a new flow settings group:

1. In FintechOS Portal, click the main menu icon at the top left corner.
2. In the main menu, select **Digital Flow Processing**.
3. Click **Flow Settings**.
4. In the Flow Settings List page, click the **Insert** button at the top right corner to add a new flow settings group.
5. In the Add Flow Settings window, enter a **Name** for your flow settings group.
6. Click the **Save and Close** button at the top right corner to save your flow settings group.

3 Add the Video Streaming automation processor to flow settings group

1. In FintechOS Portal, click the main menu icon at the top left corner.
2. In the main menu, select **Digital Flow Processing**.
3. Click **Flow Settings**.
4. In the Flow Settings List page, double click the flow settings group you wish to host your automation processor.
5. In the Edit Flow Setting window, under the Processor Settings section, click the **Insert** button to add a new automation processor.

The Add Processor Settings page appears. In this page, configure the video streaming automation processor settings as described in the next step.

4 Configure the automation processor's settings

1. In the **Add Processor Settings** screen, fill in the following fields:
 - Name – Enter a name for your automation processor
 - Digital Processor Type – Select **Video Streaming**.
 - Settings – JSON code for the automation processor's settings. For details, see ["Video Streaming Settings" below](#).
2. Click the **Save and Close** button at the top right corner to save your automation processor.

Video Streaming Settings

The Video Streaming settings are defined in JSON format as key-value pairs. The following settings are available:

Setting	JSON Key	Description
Workflow entity	SourceEntityName	Name of the entity on which you created the user journey.

Setting	JSON Key	Description
Populated entity	DestinationEntityName	Name of the entity that is populated with the data returned by video streaming.
Name	SourceLookupDestinationName	Name of the SourceEntityName lookup key that points to DestinationEntityName. If they are the same entity, enter the primary key.
	use_EU_proxy	If set to True, video streaming is done through Vonage.
	QueueParameters	<p>Defines a video streaming queue. Select a video queue either by ParamName and values, ParamValue:</p> <ul style="list-style-type: none"> • ParamName – the attribute from FTOS_DFP_QUEUE that will identify the process queue • ParamValue - the value stocked on the attribute above that will unique identify the queue <p>For example, if you want to have all video calls in a queue, then the Queue Parameters will be VideoQ (or any other name). If you want to have video calls coming from a flow which has this automation processor, to be added to a VIPQ, then set the Queue Parameters VIPQ and the video calls will enter this queue instead of the generic one, which in this case is VideoQ.</p>

Setting	JSON Key	Description
Redirect in case of success	maskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful video streaming process:</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form. <p>The section can be specified using the name of the section or the section’s number. Example "section": "Personal Data" and "section": "1" are both valid.</p>
Redirect in case of failure	maskNextStepUrlFail	<p>Location in the user interface where the user is redirected If the process did not successfully pass:</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. section – Optional parameter for the section name of the above form. <p>The section can be specified using the name of the section or the section’s number. Example "section": "Current Account" and "section": "4" are both valid.</p>

HINT
 The Video Streaming processor does not have mappings because the operator

decides if the application is approved or not, not a value stored in an attribute in the source entity.

Examples

Video Streaming settings

```
{
  "DestinationEntityName": "FTOS_BNKAP_
RetailApplicantData",
  "SourceEntityName": "FTOS_BARET_AccountApplication",
  "SourceLookupDestinationName": "retailApplicantId",
  "maskNextStepURLSuccess": { "entity":"FTOS_BARET_
AccountApplication", "form": "FTOS_BARET_AccountApplication_
UserJourney", "section": "Contract " },
  "maskNextStepURLFail": { "entity":"FTOS_BARET_
AccountApplication", "form": "FTOS_BARET_AccountApplication_
UserJourney", "section": "Video Call Unsuccessful " },
  "QueueParameters":
  [
    {
      "ParamName": "Name",
      "ParamValue": "bankAgencyA"
    }
  ]
}
```

Adding Video Streaming to a Digital Journey

Using this micro-service in a form driven flow renders the trip to the branch redundant. A customer no longer has to go to a physical store to contract a service. By simply clicking on a icon to launch the journey, and during some point the user has a call with the operator to approve or reject the application. Follow the steps to integrate the automation processor into a flow:

1. In Innovation Studio, create a custom journey to define a button to call the Video Streaming automation processor. For information on how to create a custom journey, see FintechOS Studio User Guide, section [Custom Flows](#).

2. Go to the form driven journey or form step on which you want to add the button to call the Video Streaming automation processor. This is the "SourceEntityName" on page 205 you specified when configuring the automation processor's settings.
3. Click the **Advanced** tab.
4. Click the **After Events** tab (for Steps, it is displayed by default, being the only tab available).
5. In the JavaScript field, provide the following code to call the video streaming processor:

```
//import customer library
var dfpHelper = ebs.importClientScript("FTOS.DFP");

//call VideoStreaming component custom form
$(".<video streaming button name>").click(function() {
    var componentName = 'FTOS_DFP_VideoStreaming';

    //name of the component
    var recordId = ebs.getCurrentEntityId();

    var p = {};
    p.flowSettingsName = formData.FlowSettings;
    p.processorSettingsType = 'VideoStreaming';
    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", p, function(e)
    {
        var processorSettingsName =
e.UIResult.Data.ProcessorSettingsName;
        dfpHelper.loadComponent(componentName,
processorSettingsName, recordId, false);

    });
});
```

6. Click the **Save and Close** button at the top right corner to save your digital journey.

Co-browsing

Co-browsing (also known as collaborative browsing) is a technology that allows peer-to-peer communication between two browsers and delivers instant contextual communication.

Co-browsing could be used in a number of digital journeys, both for banking and insurance use cases:

- Customer onboarding;
- Loan application;
- Mortgages;
- Compliance related processes;
- Claims handling and many more.

With Co-browsing, customers and operators can interact in real-time, streamlining an in-person experience. Operators can see the customers' screens in real-time and guide them remotely through forms, transactions, and processes by either highlighting relevant areas on the customers' screens or by taking control of the customers' web sessions and performing actions on behalf of the customers (if the customer allows it).

Co-browsing uses WebRTC (Web Real Time), an open framework for the web that enables Real-Time Communications (RTC) capabilities in the browser. This provides a 100% web-based experience that requires no downloads, installations, or plugins. For additional information on system requirements for components that use WebRTC (Web Real Time), see the [System Requirements](#) page from the Administration Guide.

FintechOS enables you to add Co-browsing to your digital journeys allowing you to connect operators and customers in real-time. This increases operators' efficiency by reducing call-handling time and provides faster customer service, enhanced customer satisfaction, and improved business performance.

Features

- **Enhanced User Experience.** High quality and scalable video communications for your FintechOS web and mobile applications;
- **Quality of Service.** Dynamically prioritize audio traffic over video on slow network connections;
- **Adaptive Layout.** Adapt stream layout and display based on audio detection;
- **Browser sharing in real time.** Operators and customers share their active web session while in a video call;
- **Visual drawing tool.** Operators can highlight a specific area on the customer's screen to instantly point the customer to specific actions;
- **Chat.** Customer and operator can interact via text messages;
- **Control switching.** The leader or the participant(s) can take control or request control of the session in seconds, and aid the other call participant by navigating on the screen in real-time.

Security

Secure Co-browsing sessions are important, especially when taking into account that personal data is exchanged in banking or insurance digital journeys. The FintechOS Co-browsing capability ensures data protection and security:

- **Data in transit is encrypted.** Co-browsing sessions use HTTPS connections, SHA-256 SSL certificate and AES 128-bit encryption to protect sensitive data in transit.
- **No data is stored.** FintechOS does not store Co-browsing session data. The data lives in the memory during the Co-browsing session.

- **Behavior control.** The customer controls the operator's permissions.
- **Isolated control.** The person who controls the session can restrict other participants' access to session secrets by using elements removal options.
- **Data Privacy.** Sensitive customer data is protected using field masking. The fields containing customer sensitive data are obfuscated, not shown in plain text to the operators and other session participants (if any were invited).
- **White/Black Listing.** Allow or deny access and privileges to specific members.
- **Action audit.** Track all actions performed during the Co-browsing session.

Installing Co-browsing

1 Install the SysPacks

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.
5. Download the **SySDigitalSolutionPackages v22.1.1000.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page,

NOTE Make sure the following packages are deployed:

- 10_01 FTOS Cognitive Processor OperatorDM - v20.2.11x1
- 10_02 FTOS Cognitive Processor OperatorScripts - 20.2.11x1

- 100 DFP Common Scripts - 21.2.11x0

2 Set up the Co-browsing Service Subscription Key

IMPORTANT!
Make sure that the Vault configurations are done and that the Webhook is created.

In Vault

Key Path	Key	Value
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServiceCobrowsingEndpoint	DCS web app endpoint URL provided by FintechOS.
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServicesCobrowsingAppld	ID for the Surfly Co-browsing service subscription.
kv/<environment>/<FintechOS Portal instance>/app-settings	FTOSServicesCobrowsingSubscriptionKey	Subscription key for the Surfly Co-browsing service.

The configuration required by Co-browsing consists of the following:

- Serilog Configurations
- EbsSqlServer: Connection string pointing to the database where configurations are stored.

In order to correctly identify the sub-account configuration, Co-browsing requires an extra configuration to be made. This helps retrieve the Surfly Api Key based on the subscription key. For this, an entry in the FintechOS management instance, the ApiKeyRelation entity has to be edited accordingly, similar to the image below:

Field	Description
ApiTypeId	The ID type. It needs to be Cobrowsing.
Name	The name given session name. Must be unique.
SubscriptionKey	Represent the display name of the given subscription key. Must be different from Name.
ApiKey	The API key to be sent to the Co-browsing session.

3 Set up the Processor Settings

Configure the following **ProcessorSetting**:

```

{
  "SourceEntityName": "Entity_Name",
  "QueueParameters": [
    {
      "ParamName": "Name",
      "ParamValue": "VideoQueue",
    }
  ]
}

```

- SourceEntityName is the entity from which the Co-browsing session is initiated.
- QueueParameters, the ParamValue is the queue value item of the Co-browsing session.

Co-browsing Streaming Flow

Co-browsing is the continuous transmission of video content between a customer and a server. The customer can start a Co-browsing session with a bank consultant or call center operator, receive real-time assistance during onboarding journeys by browsing and filling out information together with the consultant or operator.

Find below the Co-browsing streaming flow:

1. The customer initiates a Co-browsing session.
2. The FintechOS gateway sends instructions to the Co-browsing Service to create a Co-browsing session.
3. The Co-browsing Service creates a session and sends the session ID back to the FintechOS gateway.
4. The FintechOS gateway creates a unique token and sends the session information to the FintechOS Portal and the customer is added to the queue.
5. When an operator picks up the customer's session from the queue, the FintechOS gateway sends the session ID and token to the customer.
6. The customer uses the token to connect to the session.
7. The Co-browsing session starts and the participants (customer and operator) can start interacting in real-time.

Example

Let's take an example of an loan origination journey. The customer, John launched the journey on his laptop at home. After going through several steps, including identity verification, John now needs to choose the best loan offer for him. However, John needs help with picking an offer. John follows the next steps:

1. From the customer journey, John initiates a Co-browsing request.
2. The bank operator picks up the Co-browsing call from the queue.
3. The bank operator then clicks **Take call** to join the Co-browsing session.

During the session, John can add more users to the session, like his wife if he decides to add a coborrower to the journey. John can also switch control to the bank operator, thus allowing them to assist by explaining each options available in the flow's step.

When given control, the bank operator gets the same permissions as the customer.

With real-time assistance from a bank operator, the customer is able to get all the information they need and complete the journey in good time.

Adding Co-browsing to a Digital Journey

The Co-browsing automation block can be added at any step of the customer journey. Follow the steps below to integrate it in a flow:

1. In **Innovation Studio**, create a custom journey to define a button to call the Co-browsing automation block. For information on how to create a custom journey, section [Custom Flows](#).
2. Go to the form driven journey or form step on which you want to add the button to call the Co-browsing automation block.
3. Click the **Advanced** tab.
4. Click the **After Events** tab (for Steps, it is displayed by default, being the only tab available).

5. In the JavaScript field, provide the following code to call the Co-browsing processor:

```
var csl = ebs.importClientScript("FTOS.DFP.Cobrowsing")
if(typeof(__surfly) === 'undefined'){

    $("#cobrowsingButton").on("click", function() {
        csl.createCobrowsingSessionEvent(
            formData,
            window.location,
            sessionStorage.B2CSessionIdParam,
            {
                settings: {
                    // these settings change depending on
                    // what entity your form edits
                    recordId:
                    formData.model.myAccountApplicationid,
                    ProcessorSettings: "CobrowsingQueue",
                    // this is just an example, your
                    // processor/flow settings
                    // can be named however you wish
                }
            }
        )
    })
}
else {
    //$("#button#cobrowsingButton").hide();
    //if the button isn't hidden, use the other selector
    $("#button#cobrowsingButton")
        $("#cobrowsingButton").hide();
}
```

6. Click the **Save and Close** button at the top right corner to save your digital journey.

IMPORTANT!

- If the journey contains multiple processors, if the Co-browsing session is closed in a step including a processor, then the flow is automatically redirected to the first step containing a processor.
- After the Co-browsing session ends, the user needs to be redirected to the step in which the session was closed. For this, the journey needs to have a Business Workflow behind it, as well as a custom code for redirecting.

Troubleshooting

- if `ebs.importClientScript` does not load in a B2C environment and you get the following error message: "clientScriptLibray not found: FTOS.DFP.Cobrowsing", then use the following workaround:
 - On the form driven flow entity call `"formScope.csl = ebs.importClientScript("FTOS.DFP.Cobrowsing")`.
 - On the button, instead of `"csl.createCobrowsingSessionEvent"` call `"formScope.csl.createCobrowsingSessionEvent"`.
- for *recordid*, instead of `formData.model.myAccountApplicationid`, you must add the Id of the entity used for launching the Co-browsing session. The following can also be used: `recordId: ebs.getCurrentEntityId()`.
- ProcessorSettings value is the name of the file configured after installation [here](#).
- In the FTOS.DFP.Cobrowsing client script library, at the `buildURLFromFormData` and `buildURLFromCustomAction` functions, for obtaining the URL you must take into account if the `location.host` has proxy or not. The initial code is:

```
location.protocol +
    "/" +
    location.host +
    "/Main" +
    continueToUrl;
```

If the environment's URL contains "proxy" or "b2cproxy", then the following must be added:

```
continueToUrl = location.protocol +
    "/" +
    location.host +
    "/proxy/Main" +
    continueToUrl;

// or
continueToUrl = location.protocol +
    "/" +
    location.host +
    "/b2cproxy/Main" +
    continueToUrl;
```

- On the environment, the IPs obtained from this [URL](#) must be whitelisted.

Document Signing

Electronic signature or e-signature technology acts as a replacement for handwritten signatures indicating the customer's acceptance of any official document. Using a qualified electronic signature, you are able to verify the authorship of a declaration in electronic data exchange over long periods of time.

The eSign component enables you to identify the person who signed the document and verify that no one tampered with the contents of the document. In addition, you can add the Digital Documents Processor to your digital journey to automatically generate customized contracts and agreements, or other essential business documents.

eSign Processor

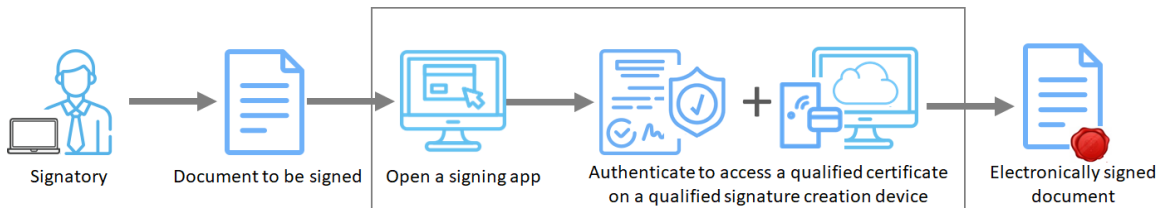
An electronic signature enables you to identify the person who signed the document and verify that no one tampered with the contents of the document. It is an electronic indication of a person's intent to agree to the content of a document or a set of data to which the signature relates.

Using a qualified electronic signature, that is an electronic signature compliant to the [eIDAS Regulation](#), you are able to verify the authorship of a declaration in electronic data exchange over long periods of time.

A qualified electronic signature ensures that:

- the data in electronic form is attached to or logically associated with electronic data and is used by the signatory to sign
- it is uniquely linked to and capable of identifying the signatory
- it is created in a way that allows the signatory to retain control

- it is linked to the document in a way that any subsequent change of the data is detectable
- it is created by a qualified signature creation device
- is based on a qualified certificate for electronic signatures.

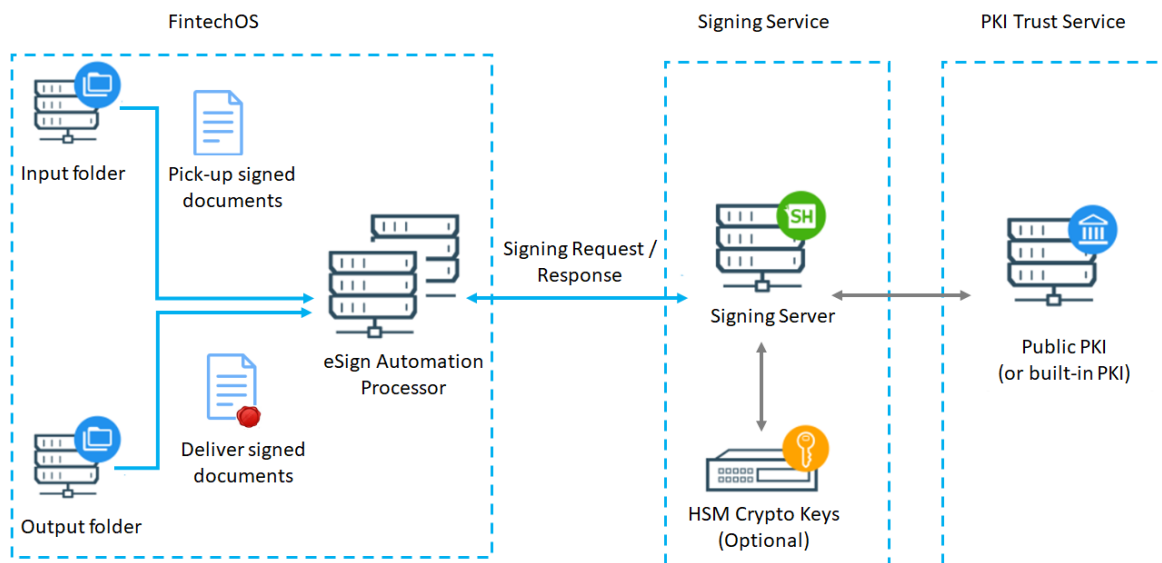


Signature creation devices, either physically owned by signatory (smartcards, SIM cards) or remote (managed by providers), protect the electronic signature creation data of the signatory. The qualified certificates and cryptographic keys for electronic signatures are provided by providers which have been granted a qualified status by a national competent authority.

You can use the eSign processor with qualified electronic signature providers.

Every time a signature is captured, a digital signature is also applied, which turns the document into a sealed, tamper-evident PDF. This tamper evidence begins when the document is initialized and maintained through document completion.

The figure below presents a high level overview of eSign architecture:



eSign Automation Processor Features

- Remote e-signing
- Image signature with Click-, Type-, or Draw-to-Sign
- Real-Time identification of the signatory
- Disposable certificates (valid for 60 min)
- Base64 signed files encoding
- Detailed audit trail of who signed the document and when, including all transactions and events performed on the document, emails and notifications sent
- Tracking the status of your signature requests
- Download envelope log.

Applications

eSign can be used to simplify paper-driven financial or insurance processes, such as:

- Customer onboarding
- Account opening
- Loan applications
- Compliance related processes
- Claims handling

Installing eSign

Install Server Configuration

First, make sure you have the right application dependencies installed and configured.

In the physical location of the site (Portal OR B2C Portal):

Portal/ B2C Portal web.config <appSettings> section, configuration

If you have a generic key for all the services, add the following keys in web.config - <appSettings>.

```
<add key="FTOSServicesEndpoint" value="get-the-url-from-portal"/>
<add key="FTOSServicesAppId" value="get-the-key-from-portal"/>
```

If you have different subscription keys for each of the service you must add the following keys in web.config - <appSettings> as following:

ESign

```
<add key="FTOSServicesESignEndpoint" value="get-the-url-from-portal"/>
<add key="FTOSServicesESignAppId" value="get-the-key-from-portal"/>
```

Install Application Configuration

Log in to Innovation Studio:

1 Import Packs

Import in Deployment Package the packs from the Pack received. The **08 eSign_Processor - vxx.x.xxxx.zip** pack contains the services configuration model and scripts for ESign.

2 Modify data

Import the **Pkg08_ESignProcessor_01_FlowSetting_Example_vx.x.xml.zip** from **..\02 ConfigurationDataDeploymentPackages** form **DevOps -> Configuration data deployment package**.

Log in to Innovation Studio:

IMPORTANT!

For version smaller than v21.1.x, the Flow Settings menu is in the FintechOS Portal: **Digital Flow Processing -> Flow Settings.**

3 ESign Configuration

For the configuration of Automation Blocks, select **Digital Flow Processing -> Flow Settings.**

In the “Example Flow Setting Esign” you can find a eSign setting and mapping examples. You can copy the examples and modify them.

Processor Settings fields

Settings

Key	Settings
DestinationEntityName	Name of destination entity
SourceEntity Name	Name of source entity; Needed only if the Face Recognition process starts from an edit form; if so, the entity name is needed to update the business status after the Face Recognition process ends.
SourceLookpDestination	Name Name of the lookup key from SourceEntityName linked to DestinationEntityName
FileAttribute NameList	The files that will be signed on that request, there can be one ore more: <ul style="list-style-type: none"> fileAttributeName : The name of the file attribute where the pdf document that will be sign is saved. fileToBeSignedName: The name of the files that will be signed.
MaxRetry	Number of Sign retries.
daysUntilExpire	Number of days from the creation of the signature request until the envelope will expire if not signed. Default is 7 days.
signedDocumentName	The name that will be given to the signed document.

Key	Settings
"WebhookUrl"	"Getting Status Changes Notifications Using Webhooks" on page 247
"WebhookStatusUrl"	"Getting Status Changes Notifications Using Webhooks" on page 247
order	The signatures workflow order -- it could be sent in parallel
signatureTag	The tag from the pdf that will be replaced with the signature field , Ex: "#esaw#"
signatureTypeTemplate	""
signatureType	<ul style="list-style-type: none"> • QualifiedElectronicSign • Click2Sign • AutomaticSign • OTPSign • NoSign • RemoteSign.
ClearSignatureString	<ul style="list-style-type: none"> • True - The signature string is cleared after a signature is applied. Implies changing the initial file which can cause issues with a document that has already been signed before the signing flow began. • False - Keeps the initial signature string.
automaticProfile	<pre>{ "attributeKey": "YourAttribute" }</pre> <p>OR</p> <pre>{ "attributeKey": null }</pre> <p>- See details here " 5 ESign Configuration for Automatic Signature Profile" on page 230</p>

Key	Settings
signatureCoordinates	<p>The configuration for the signature that will be placed by coordinates:</p> <pre>[{ "fileName": "pictureOCR", //the file attribute that contains the file that will be signed by coordinates "pageNumber": 1, // the page number where the signature will be placed "x": 250, // the x coordinate on the page "y": 80 // the y coordinate on the page }]</pre>
OTP_SMS_S howNoInSignature	<p>true : will display the phone number on the signature.</p> <p>false : will not display the phone number on the signature.</p>
languageCode	The language of the document.
email	The email of the person that will sign.
phoneMobile	<p>The Phone Number of the person that will sign OTP Sign:</p> <ul style="list-style-type: none"> - If set, the signature won't ask for your phone number, will send the OTP to this phone. - If not set (send null or does not exist in list), the recipient will be asked to enter the phone number where the OTP will be sent.
prefixPhone Mobile	Prefix of the phone number.
firstName	The first name of the person that will sign.
lastName	The last name of the person that will sign.
countryResidence	<p>The country of Residence of the person that will sing</p> <p>Not mandatory for OTP Sign .</p>
documentType	<p>The type of document used for signing ("CI")</p> <p>Not mandatory for OTP Sign.</p>
documentIssuedBy	<p>The institution that issued the ID/ Pass.</p> <p>Not mandatory for OTP Sign.</p>

Key	Settings
socialSecurityNumber	The Social Number of the client. Not mandatory for OTP Sign.
documentExpiryDate	The expiration date of the ID/ Pass. Not mandatory for OTP Sign.
documentIssuedOn	The issue date of the ID/ Pass Not mandatory for OTP Sign.
documentNumber	The document number of the ID Not mandatory for OTP Sign
smsText	The text that the client will receive. Not mandatory for OTP Sign.
clickMsg	The Text displayed on the place where the client will sign. Not mandatory for OTP Sign.
maskNextStepURLSuccess	Information used to create the link loaded after the Face Recognition process ends successfully.
entity	Entity name.
form	Form name.
section	Optional field, can be either the section number or the section name.
maskNextStepURLFail	Information used to create the link loaded after the OCR process ends with errors.
entity	Entity name.
form	Form name.
section	Optional field, can be either the section number or the section name.
businessStatusSuccess	Business status name applied to SourceEntity if Face Recognition process ends successfully; Needed only if the Face Recognition process runs on form in edit mode.
businessStatusFail	Business status name applied to SourceEntity if Face Recognition process ends with errors; Needed only if the Face Recognition process runs on form in edit mode.

Mappings

Key	Settings
FileAttributeName	
Map	Area of actual mappings.

Key	Settings
key	Name of the field as received from Face Recognition Process.
value	Name of the field as declared in the entity where the result is saved.

ESign Load component examples

Example for Example Flow Setting: Javascript code - After Events:

```

var dfpHelper = ebs.importClientScript('FTOS.DFP');

var componentName = "FTOS_DFP_ESign"; //name of the
component
var recordId = ebs.getCurrentEntityId();
var fileExists = true; //"documentToBeSigned" argument =
source entity file attribute

var flowSettingsName = formData.FlowSettings;

var p = {};
p.flowSettingsName = formData.FlowSettings;
p.processorSettingsType = 'ESign';
p.processorSettingsName = 'ESign_Example';

ebs.callActionByName("FTOS_DFP_FlowProcessorSettingsByType",
p, function(e)
{
    var processorSettingsId =
e.UIResult.Data.ProcessorSettingsId;
    dfpHelper.loadComponent(componentName,
processorSettingsId, recordId, fileExists);
});

```

NOTE

The Result of ESign execution will be saved in this entity: **FTOS_DFP_ESign**.

4 ESign Download Configuration

For the configuration of Esign, select **Digital Flow Processing -> Flow Settings**.

In “Example Flow Setting Esign” you can find a Esign setting and mapping examples. You can copy the examples and modify them.

Processor Settings fields

Settings

Key	Settings
EntityName	The entity that will have the file saved to.
ESignFlowSettingName	The name of the Flow Setting used to host the processor for the eSign request.
ESignProcessorName	The name of the processor setting used to send sign request.
maskNextStepURLSuccess	<pre>{ "entity": "NemOf entity", "form": "Name of UserJourney", "section": "NameOfSuccessSection" },</pre> <p>The step that will be displayed if the download is successful.</p>
maskNextStepURLFail	<pre>{ "entity": "NemOf entity", "form": "Name of UserJourney", "section": "NameOffailSection" },</pre> <p>The step that will be displayed if the download has failed.</p>

Mappings

Key	Settings
DownloadFilesMapping	The list of the files that will be downloaded.
FileAttributeName	Where will the file be downloaded to, the name of the attribute.
FileName	The name that will the downloaded file will have after download.

ESign Load component examples

Example Flow Setting: Javascript code - After Events:

```

var dfpHelper = ebs.importClientScript('FTOS.DFP');
var componentName = "FTOS_DFP_ESign_Download"; //name of the
component
var recordId = ebs.getCurrentEntityId();
var fileExists = true;

var flowSettingsName = formData.FlowSettings;
var p = {};
p.flowSettingsName = formData.FlowSettings;
p.processorSettingsType = 'ESign';

var businessStatusName = ebs.getFormData
().model.businessStatus.name;

if (businessStatusName == "Documents Signed") {
    p.processorSettingsName = 'ESign_Download_Documents';
}
else if (businessStatusName == "Contract Signed") {
    p.processorSettingsName = 'ESign_Download_Documents';
}
console.log("p.processorSettingsName "+
p.processorSettingsName);
if (p.processorSettingsName) {
    ebs.callActionByName("FTOS_DFP_
FlowProcessorSettingsByType", p, function (e) {
        var processorSettingsId =
e.UIResult.Data.ProcessorSettingsId;
        dfpHelper.loadComponent(componentName,
processorSettingsId, recordId, fileExists);
    });
}

```

5 ESign Configuration for Automatic Signature Profile

The automatic signature has a key value that must be set, this key value is the form the Namirial Account that has activated the automatic signature option.

This key value is inserted in the table: **FTOS_DFP_ESignKey**

Name and Key Type are not relevant for the flow to work, but will help you organize your records.

The most important attribute is **Key Value** and is the value described above.

The key is provided by Namirial when you configure the **Automatic Remote Signature Profiles**. More than one profiles can be set, depends on the need.

Auto Profile Case 1

If there is only one automatic profile set in Namirial then you can fill in only the **Name** and **Key Value** for that profile.

In The Processor Setting field the configuration will look like this:

```

"signatureType": "AutomaticSign",
  "automaticProfile": {
    "attributeKey": null
  }
    
```

Auto Profile Case 2

If there are more than one automatic profiles set in Namirial then will can fill in only the **Name, Key Record Id and Key Value** for that profile. The Key type is for you to fill in to know from where that Key Record Id is from. (In the Key Type you can fill in **systemuser**)

In The Processor Setting field the configuration will look like this:

```
"signatureType": "AutomaticSign",
  "automaticProfile": {
    "attributeKey": "userId"
  }
```

For example:

If you have more users from FintechOS with the right to sign automatically, then the configuration is like this:

userId: will be then name of the attribute from your entity that stores the userid from FintechOS.

In **FTOS_DFP_ESignKey**, the **Key record id** will be the id of the **userid** form the **systemuser** table.

Auto Profile Case 3

You can set other table (**example CustomProfileTable**) for your profiles, not system user, this means that **Key record id** will be the id from the record form that table and **yourAttribute_From_EntityName:** will be then name of the attribute from your entity that stores the Record ID from the table created.

(In the Key Type you can fill in **CustomProfileTable**)

Example:

```
"signatureType": "AutomaticSign",
  "automaticProfile": {
    "attributeKey": "yourAttribute_From_EntityName"
  }
```

yourAttribute_From_EntityName= CustomProfileTableId (the lookup from your entity to the CustomProfileTable for example or just an attribute that stores that RecordId).

This way the system will know witch Automatic Profile to use to sign.

Upgrade Application Configuration

1 Import Again Packs

Import in **Deployment Package** the packs from the Pack received.

2 Modify data**IMPORTANT! |**

Using the example, please adapt your processors settings with the new structure found in the new examples.

The **Pkg08_ESignProcessor_01_FlowSetting_Example_v1.0.xml.zip** from **..\02 ConfigurationDataDeploymentPackages** from **DevOps -> Configuration data deployment package**.

Setting Up an eSign Automation Processor

The list of available signatures are:

- qualified electronic signature
- automatic signature
- OTP
- Click2Sign
- RemoteSign.

Step 1. Create a digital flow processing settings group

The Automation Blocks automation processor must be hosted inside a flow settings group. A flow settings group can include multiple automation processors and is typically used as a container for the automation processors called by a specific digital journey.

If you already have a flow settings group you wish to host your eSign automation processor, skip to "[Step 2. Add the eSign automation processor to a generic processor settings group](#)" on the next page. Otherwise, follow the instructions below to create a new flow settings group:

1. In FintechOS Portal, click the main menu icon at the top left corner.
2. In the main menu, select **Digital Flow Processing**.

3. Click **Flowr Settings**.
4. In the Flow Settings List page, click the **Insert** button at the top right corner to add a new flow settings group.
5. In the Add Flow Settings window, enter a **Name** for your flow settings group.
6. Click the **Save and Close** button at the top right corner to save your flow settings group.

Step 2. Add the eSign automation processor to a generic processor settings group

1. In Innovation Studio, click the main menu icon at the top left corner.
2. In the main menu, select **Digital Flow Processing**.
3. Click **Flowr Settings**.
4. In the Flow Settings List page, double click the flow settings group you wish to host your automation processor.
5. In the Edit Flow Setting window, under the Processor Settings section, click the **Insert** button to add a new automation processor.

The Add Processor Settings page appears. In this page, configure the video streaming automation processor settings as described in the next step.

Step 3. Configure the automation processor's settings

1. In the **Add Processor Settings** screen, fill in the following fields:
 - Name – Enter a name for your automation processor
 - Digital Processor Type – Select **Electronic Signature**.
 - Settings – JSON code for the automation processor's settings. For details, see ["eSign Settings" on the next page](#).

- Mapping – JSON code for the automation processor's mappings. For details, see ["eSign Mappings" on page 238](#).

2. Click the **Save and Close** button at the top right corner to save your automation processor.

eSign Settings

The eSign settings are defined in JSON format as key-value pairs. The following settings are available:

JSON Key	Description
SourceEntityName	The entity associated with the business workflow (digital journey) that calls the ESign process. Needed only if the ESign process is used on an edit form (to alter an existing record) to update the workflow entity's business status after the signing (see "businessStatusSuccess" on page 237 and "businessStatusFail" on page 237).
DestinationEntityName	The name of the entity on which the E-sign results are mapped.
SourceLookupDestinationName	The name of the SourceEntityName lookup key that points to DestinationEntityName. This is the source entity of the user journey from which the automation processor gathers data. If they are the same entity, enter the primary key.
fileAttributeName	The name of the attribute on the entity which will store the file to be signed.
maxRetry	The maximum number of signing attempts. If this number of failed signing attempts is reached, the user will be redirected according to the specifications in the "maskNextStepUrlFail" on page 237 .
SignedDocumentName	The name of the attribute which will store the signed file.
order	The order of the signatures from the workflow. They can be sent in parallel.

JSON Key	Description
signatureTag	Marks the place where the signature will be done in the PDF file. It is a tag within the document to be signed on which customers will click to sign.
signatureTypeTemplate	The signature type. This setting cannot be null, but can be an empty string if there is no specific signature type required.
checkCertificate	Checks whether or not the client has a valid certificate.
fileToBeSignedName	The name of the file that has to be signed.
languageCode	The code of the language used for the signature. For information on how to add a new language, see the Innovation Studio User Guide, section <i>Adding New Languages</i> .
email	The attribute that stores the email provided by the customer. The email address of the customer where the document to be signed will be sent to.
phoneMobile	The attribute which stores the phone number provided by the customer. It is the phone number that will receive the "textMessage" provided in the "smsText" on the next page attribute.
prefixPhoneMobile	The attribute that stores the country prefix of the mobile phone number provided by the customer.
firstName	First Name of the customer.
lastName	Last Name of the customer.
countryResidence	The attribute which stores the country code where the customer has residence.
documentType	The type of the document processed by the OCR processor if customer data has been gathered using an OCR automation processor.
documentIssuedBy	Country/Institution that issued the document.
socialSecurityNumber	The name of the attribute that maps the customer's social security number. For example, CNP for Romania and SSN for other countries. If on an entity, this info is stored in an attribute called PIN, then put PIN here.
documentExpiryDate	The name of the attribute which stores the expiration date of the contract.
documentIssuedOn	The name of the attribute which stores date when the document to be signed is issued.
documentNumber	The name of the attribute which stores the number of the identity document.

JSON Key	Description
smsText	The name of attribute that stores the text that is sent to the specified mobile phone number.
clickMsg	The name of the attribute that stores the message displayed to the person that is responsible to sign the document.
maskNextStepUrlSuccess	<p>Location in the user interface where the user is redirected after a successful scan.</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. step – Optional parameter for the step name of the above form. <p>The step can be specified using the name of the step or the step’s number. Example "step": "Personal Data" and "step": "1" are both valid.</p>
maskNextStepUrlFail	<p>Location in the user interface where the user is redirected after the maximum number of failed scan attempts (see "maxRetry" on page 235).</p> <ul style="list-style-type: none"> entity – Entity name. form – Form name of the above entity. step – Optional parameter for the step name of the above form. <p>The step can be specified using the name of the step or the step’s number. Example "step": "Personal Data" and "step": "1" are both valid.</p>
businessStatusSuccess	Indicates that the contract has been signed.
businessStatusFail	Indicates that the contract has been rejected.

Examples

eSign settings

```
settings{
```

```

        "DestinationEntityName": "FTOS_BNKAP_
RetailApplicantData",
        "SourceEntityName": "FTOS_BARET_AccountApplication",
        "SourceLookupDestinationName": "retailApplicantId",
        "FileAttributeName": "contract",
        "MaxRetry": 3,
        "signedDocumentName": "signedContract",
        "order": "1",
        "signatureTag": "#esaw#",
        "signatureTypeTemplate": "",
        "fileToBeSignedName": "esign.pdf",
        "languageCode": "EN",
        "email": "email",
        "phoneMobile": "mobilePhone",
        "prefixPhoneMobile": "+4",
        "firstName": "firstName",
        "lastName": "lastName",
        "countryResidence": "RO",
        "documentType": "CI",
        "documentIssuedBy": "IdIssueInstitution",
        "socialSecurityNumber": "PIN",
        "documentExpiryDate": "IdExpirationDate",
        "documentIssuedOn": "IdIssueDate",
        "documentNumber": "IdCardSeries",
        "smsText": "Acordul pentru semnarea documentului
prin intermediul certificarii digitale, codul tranzactiei:
{tId}",
        "clickMsg": "Click here to sign",
        "maskNextStepURLSuccess": { "entity": "FTOS_BARET_
AccountApplication", "form": "FTOS_BARET_AccountApplication_
UserJourney", "section": "Contract Successful " },
        "maskNextStepURLFail": { "entity": "FTOS_BARET_
AccountApplication", "form": "FTOS_BARET_AccountApplication_
UserJourney", "section": "Contract Fail " },
        "businessStatusSuccess": "Contract Signed",
        "businessStatusFail": "Contract Rejected"
    }

```

eSign Mappings

The eSign mapping match the field name as returned by the ESign (keys) with the name of the document that needs to be signed.

Setting Name	Description
fileAttributeName	The name of the document that needs to be signed.

Examples

Sample JSON code for eSign mapping

```
{
  "fileAttributeName": "contractSigned"
}
```

E-sign with tags or coordinates or both

To configure the signing of a document that was either configured with FintechOS' [Digital Documents Processor](#) or a document generated externally by a bank of insurance company, it is possible to sign it by placing a tag or with coordinates or both on a PDF file by creating a request in two ways:

- in FintechOS directly using workflow library FTOSServices by calling the getWorkstepUrl
- by calling the url in Azure directly without going to FintechOS.

Using workflow library FTOSServices

The request can look like this:

```
{
  "Authentication": {
    "Username": "john.smith@company.com",
    "Key": "72b2f349-b5c7-4df7-847d-efd8a7f4a6c4"
  },
  "SignedDocumentName": "signedContract",
  "WorkstepConfigs": [
    {
      "SignatureTag": "#esaw#",
      "SignatureType": "1",
      "SignatureCoordinates": [{
        "FileName": "testR_fa303a83-6f33-49d5-b747-f4bcd2aed258_09dfefbd-296d-4247-aa1b-b4811a557632.pdf",
        "PageNumber": 1,
        "X": 225.23,
        "Y": 225.23
      }],
      "SignatureTypeTemplate": "",
      "Recipient": {
        "LanguageCode": "RO",

```

```

    "Email": "andrei.dana@fintechos.com",
    "FirstName": "Maria",
    "LastName": "Conache",
    "CountryResidence": "RO",
    "PhoneMobile": "+40727776494",
    "DocumentType": "CI",
    "DocumentIssuedBy": "RO",
    "SocialSecurityNumber": "2920530090075",
    "DocumentExpiryDate": "2024-05-30",
    "DocumentIssuedOn": "2017-05-22",
    "DocumentNumber": "IF557234"
  },
  "ClientActionUrl": "https://www.google.ro",
  "SmsText": "<TransactionCodeConfiguration
trConfId='disposableCertificateEnrolAndSignSmsText'
language='ro'><Message>Prin acest cod iti exprimi acordul pentru
semnarea documentului prin intermediul certificarii digitale,codul
tranzactiei:
{tId}</Message><hashAlgorithmIdentifier>Sha256</hashAlgorithmIdentifi
fier></TransactionCodeConfiguration>",
  "ClickMsg": "Click pt semnare",
  "Order": 1,
  "SignatureProperties": {
    "Width": 120.0,
    "Height": 80.0
  },
  "SubTaskList": [{
    "Id": "ann1",
    "DisplayName": "Anotare display",
    "AdditionalInfo": "Anotare ann",
    "TextConfig": {
      "DocRefNumber": 1,
      "PageNumber": 1,
      "TextAlign": "Left",
      "PositionX": "100",
      "PositionY": "100",
      "Editable": "0",
      "DefaultText": "##ClientTime##",
      "Format": "dd/MM/yyyy",
      "FontId": "FtosFontId1"
    }
  }
}
}
}
{
  "SignatureTag": "#esawo#",
  "SignatureType": "1",

```



```

    "SignatureCoordinates": [{
      "FileName": "testFileName.pdf",
      "PageNumber": 1,
      "X": 325.15,
      "Y": 325.15
    }],
    "Recipient": {
      "LanguageCode": "RO",
      "Email": "andrei.dana87@gmail.com",
      "FirstName": "Maria",
      "LastName": "Conache",
      "CountryResidence": "RO",
      "PhoneMobile": "+40727776494",
      "DocumentType": "CI",
      "DocumentIssuedBy": "RO",
      "SocialSecurityNumber": "2920530090075",
      "DocumentExpiryDate": "2024-05-30",
      "DocumentIssuedOn": "2017-05-22",
      "DocumentNumber": "IF557234"
    },
    "ClickMsg": "Apasa aici",
    "ClientActionUrl": "https://showcase.fintech-os.com/dcs",
    "SmsText": "<TransactionCodeConfiguration
trConfId='disposableCertificateEnrolAndSignSmsText'
language='ro'><Message>Prin acest cod iti exprimi acordul pentru
semnarea documentului prin intermediul certificarii digitale,codul
tranzactiei:
{tId}</Message><hashAlgorithmIdentifier>Sha256</hashAlgorithmIdentifi
fier></TransactionCodeConfiguration>",
    "Order": 2,
    "SignatureProperties": {
      "Width": 120.0,
      "Height": 80.0
    }
  }
],
"Files": [
  {
    "Name": "testFileName.pdf",
    "FtosFile": "
[{\ "Name\":"test.pdf\","RealName\":"test.pdf\"}]",
    "Base64Content": "base64StringWithTheFile"
  }
]
}

```

In order to correctly identify where we are adding the signature via coordinates, the `SignatureCoordinates` object contains:

- `FileName`: this needs to be the same as the `Name` property of the file in the `Files` array of the given request
- `PageNumber`: the number of the page on which we wish to add the signature in the chosen file
- `X` and `Y`: coordinates of where to put the signature. The 0,0 coordinate of the file is in the bottom left corner. The `X` and `Y` are the coordinates that start from the bottom left corner of the PDF file.

Click2Sign

This type of signature available offers a quick method of digital signature by presenting a button with a message configured in the request, a URL where the user is taken after signing and the properties of the signature itself. Once clicked, unlike the OTP type, there is no SMS sent. The location of the signature is configured placing a tag or with coordinates or both on a PDF file by creating a request.

Add the following work step in the request.

```
{
  "SignatureTag": "#esaw#", //the tag from the document where
  you want to place a signature field
  "SignatureType": "2", //2 - for Click2Sign
  "Recipient": {
    "LanguageCode": "RO",
    "Email": "joe.doe@fintechos.com",
    "FirstName": "Joe",
    "LastName": "Doe",
    "CountryResidence": "RO",
    "PhoneMobile": "",
    "DocumentType": "",
    "DocumentIssuedBy": "",
    "SocialSecurityNumber": "",
    "DocumentExpiryDate": "",
    "DocumentIssuedOn": "",
    "DocumentNumber": ""
  },
}
```

```

    "ClickMsg": "Click Here", //the message from the signature
    field, before signing
    "ClientActionUrl": "url_for_redirect_after_signing",
    "SmsText": "",
    "Order": 2, //the order of the signatures from the whole
    request
    "SignatureProperties": {
        "Width": 120.0,
        "Height": 80.0
    }
}

```

RemoteSign

This type of signature available offers access to remote certificates such as Advanced Electronic Signature or Qualified Electronic Signature to sign the document.

Add the following work step in the request.

```

{
    "SignatureTag": "#esawo#",
    "SignatureType": "5",
    "Recipient": {
        "LanguageCode": "RO",
        "Email": "maria.conache@fintechos.com",
        "FirstName": "Maria",
        "LastName": "Conache"
    },
    "ClickMsg": "Apasa aici",
    "ClientActionUrl": "https://showcase.fintech-os.com/dcs",
    "SmsText": "<TransactionCodeConfiguration
trConfId='disposableCertificateEnrolAndSignSmsText'
language='ro'><Message>Prin acest cod iti exprimi acordul pentru
semnarea documentului prin intermediul certificarii digitale,codul
tranzactiei:
{tId}</Message><hashAlgorithmIdentifier>Sha256</hashAlgorithmIdenti
fier></TransactionCodeConfiguration>",
    "Order": 1,
    "SignatureProperties": {
        "Width": 120.0,
        "Height": 80.0
    }
}

```

The following steps are taken to use the RemoteSign capability:

Configurations for signature when the sender has not filled in the dates (if the sender defines the user ID, the user has to sign without selecting the ID) :

- Clicking on the remote signature field.
- Filled in the user Id, select the device ID from the drop-down list and request a OTP.

Adding eSign to a Digital Journey

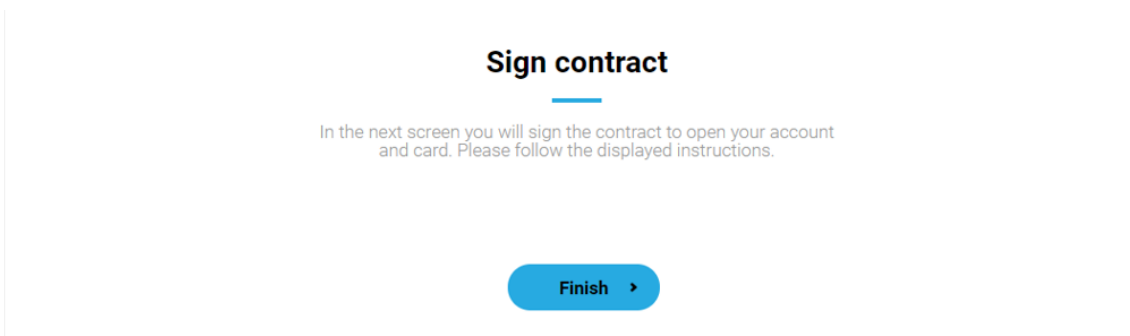
1. In Innovation Studio, create a custom journey to define a button to call the eSign automation processor. For information on how to create a custom journey, see Innovation Studio User Manual, section *Creating Custom Journeys*.
2. Go to the form driven journey or form step on which you want to add the button to call the eSign automation processor.
3. Click the **Advanced** tab.
4. Click the **After Events** tab (for Steps, it is displayed by default, being the only tab available).
5. In the JavaScript field, provide the following code:

```
//import client library
var dfpHelper = ebs.importClientScript("FTOS.DFP");
//call ESign component custom form
var componentName = "FTOS_DFP_ESign"; //name of the component
var recordId = ebs.getCurrentEntityId();
var fileExists = true; /"document" argument = source entity file attribute.It only contains the true value considering that the document already exists
var p = {};
p.flowSettingsName = formData.model.FlowSettings;
p.processorSettingsType = 'ESign';
ebs.callActionByName("FTOS_DFP_FlowProcessorSettingsByType",
p, function(e) {
    var processorSettingsName =
e.UIResult.Data.ProcessorSettingsName; //configuration of the component
```

```
dfpHelper.loadComponent(componentName,  
processorSettingsName, recordId, fileExists);  
});
```

6. Click the **Save and Close** button at the top right corner to save your form driven journey (If you're in a form step, save the step, then save the form driven journey)

This is what the customers see in the user journey:

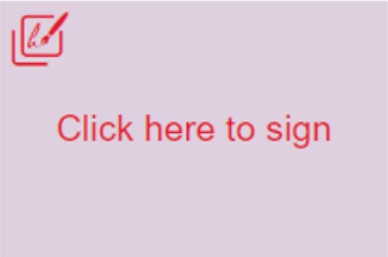


After they click **Finish**, they will be prompted to accept request for signature.



The contract/agreement pdf will be displayed. On the last page of the document, the customers will have to sign the document by clicking the button displayed next to the Account Holder Signature.

Account Holder Signature:



Once they sign it, the customer will have to accept the terms and agreements.

Issuance of disposable certificate and signature

* "Holder": the "Signatory", which is a natural person who creates an Electronic Signature;
* "Interested Third Party": the legal or natural person that gives consent to the issue of the certificates belonging to the owner of their organization, as well as representative powers, titles or appointments vested in the latter;
* "Local Registration Authority (IRA)": the legal or natural person, authorized by Namirial to carry out operations of issuing of Certificates;
* "Identification and Registration Operations": the activities of identification and registration of the Holder, in accordance with the procedures set out in the Operative Manual, in the CPS, in the Term and Conditions for Use and art. 24.1 of eIDAS;
* "Electronic Signature": means data in electronic form which is attached to or logically associated with other data in electronic form and which is used by the Signatory to sign;
* "Digital Signature": is a particular type of "Qualified Electronic Signature" based on a system of related cryptographic keys (one of them private and one public) ensuring its holder (through the Private Key) and the receiver (through the Public Key), to make the origin and the integrity of an e-document or documents clear and to verify such origin and integrity. The validity of the Digital Signature is equivalent to that of a handwritten signature;
* "Public Key": the element of the related cryptographic keys which is to be made public, with which the Digital Signature affixed to an Electronic Document of the Holder;
* "Private Key": the element of the related cryptographic keys, which is known only to the Holder, through which the Digital Signature is affixed to the Electronic Document;
* "QSCD": qualified electronic signature creation device means an electronic signature creation device that meets the requirements laid down in Annex II of eIDAS;
* "Authentication Credentials": the code or codes to identify the Holder, which are known exclusively to the latter for the use of the Certificate on Electronic Documents;
* "E-Mail Address": the electronic address provided by the Holder to which the Certification Service Provider will send all communications relating to the Contract as defined at Art. 2;

In order to sign with electronic signature the Application Form, you need to accept the terms and conditions by checking the 3 boxes below:

- *(1) I certify the content of SECTION E - SELF CERTIFICATION BY THE HOLDER
- *(2) I accept the General Terms and Conditions (Mod.NAMCA01D) and the one-sided clauses set forth in SECTION F - ONE-SIDED CLAUSES
- *(3) I give the consent to the processing of personal data as set forth in SECTION G - CONSENT TO THE PROCESSING OF PERSONAL DATA

Once they've given their consent for electronic signature and data processing they will have to click **Accept**. They will receive an OTP code via SMS.

Issuance of disposable certificate and signature

Please insert the OTP for the transaction '8jQPuAnWLj' to issue a disposable certificate and digitally sign the document.

OTP

Your secret code will expire in 4:16

CANCEL

CONFIRM

Once they provide the received OTP code within the code validity timeframe and click **Confirm**, the document is successfully e-signed and returned to the FintechOS platform in real-time.

Getting Status Changes Notifications Using Webhooks

Webhooks allow you to get programmatical notifications from FintechOS about status changes of your envelope as they happen.

For every envelope event, webhooks will send the notifications as an HTTP POST request, with a JSON body, to the endpoint you specify. They will push information to your endpoint.

Using FintechOS webhooks you can receive notification for the following events:

- the client opened the document
- the signature for client is finished
- the whole document is signed

To Get Envelope Status Changes Notifications Using Webhooks

Step 1. Configure webhooks

Prerequisite: You need to know the webhook ID received from FintechOS.

To get notifications on envelope status changes using webhooks, in the sign request, add the webhook finish URL (WebhookUrl) and webhook for status URL (WebhookStatusUrl).

Example

In this example, we set notifications for all events that occur on the document 'signedContract'.

```
{
  "Authentication": {
    "Username": "john.doe@fintechos.com",
    "Key": "72b2f349-b5c7-4df7-478d-efd8a7f4a6c9"
  },
  "SignedDocumentName": "signedContract",

  "WebhookUrl": "{FTOSHook}/hookId_
receivedFromFTOS?test=12345&asdf=abc&envelopeId=##EnvelopeId
##&recipientEmail=##RecipientEmail##&recipientOrder=##Recipi
entOrder##&action=##Action##",
  "WebhookStatusUrl": "{FTOSHook}/hookId_
receivedFromFTOS?test=12345&asdf=abc&envelopeId=##EnvelopeId
##&recipientEmail=##RecipientEmail##&recipientOrder=##Recipi
entOrder##&action=##Action##",

  "WorkstepConfigs": [
    {
      ...
    }
  ]
}
```

Where:

- **{FTOSHook}** is the webhook ID received from FintechOS. It is a constant value for all the requests from your app. In the query parameters of the WebHookUrl (after the ? mark) you can add custom values, for example test=12345 and you will receive the same value on the notification.
- **WebhookUrl** will be called only when the envelope is finished (all worksteps have been completed) and you can download the signed document.
- **WebhookStatusUrl** will be called for every status change of the envelope.

Step 2. Create an endpoint for the webhook

Create the endpoint (action) that contains the logic for receiving the webhook. In most of the cases, in the server automation scripts, call the request **downloadSignedDocument** and get the intermediate URL for the second workstep or download the signed document (the pdf file).

The example request provided at Step 1 returns the following data in the server automation script:

WebhookStatusUrl Return Example

```
{
  "UserId": "4afdc8a9-eb91-4359-81d6-c3a462fae866",
  "Ids": null,
  "Id": "00000000-0000-0000-0000-000000000000",
  "EntityName": null,
  "PreviousBusinessStatus": null,
  "PreviousBWS": null,
  "BusinessStatus": null,
  "CurrentBWS": null,
  "Values": null,
  "AdditionalValues": null,
  "PropertyBag": {},
  "ExecutionDepth": 0,
  "MessageSuccess": false,
  "Message": null,
  "ReloadPage": false,
  "NavigateToUrl": null,
  "NavigateToEntityPage": false,
  "NavigateToEntityPageOnEdit": false,
  "NavigateToEntityName": null,
  "NavigateToEntityFormName": null,
  "NavigateToEntityId": null,
  "NavigateToEntityInsertDefaults": null,
  "Data": {
    "test": "12345",
    "asdf": "abc",
    "envelopeId": "7bfa100a-f52c-4915-b25d-210e6bb82f48",
    "recipientEmail": "",
    "recipientOrder": "",
    "action": "workstepOpened"
  },
}
```

```

    "Fetch": null,
    "BeforeValues": null,
    "MergedValues": {}
  }

```

WebhookUrl Return Example

```

{
  "UserId": "4afdc8a9-eb91-4359-81d6-c3a462fae866",
  "Ids": null,
  "Id": "00000000-0000-0000-0000-000000000000",
  "EntityName": null,
  "PreviousBusinessStatus": null,
  "PreviousBWS": null,
  "BusinessStatus": null,
  "CurrentBWS": null,
  "Values": null,
  "AdditionalValues": null,
  "PropertyBag": {},
  "ExecutionDepth": 0,
  "MessageSuccess": false,
  "Message": null,
  "ReloadPage": false,
  "NavigateToUrl": null,
  "NavigateToEntityPage": false,
  "NavigateToEntityPageOnEdit": false,
  "NavigateToEntityName": null,
  "NavigateToEntityFormName": null,
  "NavigateToEntityId": null,
  "NavigateToEntityInsertDefaults": null,
  "Data": {
    "test": "12345",
    "asdf": "abc",
    "envelopeId": "7bfa100a-f52c-4915-b25d-210e6bb82f48",
    "recipientEmail": "",
    "recipientOrder": "",
    "action": "envelopeFinished",
    "FTOSIsFinishCallback": "1"
  },
  "Fetch": null,
  "BeforeValues": null,
  "MergedValues": {}
}

```

Download Envelope Log

The fourth pillar of security, logging provides you with comprehensive audit trail of what happened at any given time and who performed the action.

NOTE

The audit trail displays all the information about the document and it can be downloaded from Namirial in PDF format together with the signed document. This is done when the property `downloadLog` is set to true. The configurations are made in the processor settings. If the processor is not used, then this field is sent in the body of the request to DCS.

The envelope log file shows information about the envelope:

- the general information
- status of the envelope
- the creation date
- the recipients and the changes they made to the document, e. g., which signature fields were placed on the document
- the time when the workstep was finished.

How to download the envelope log

To download the envelope log:

1. Log in the Innovation Studio in developer mode.
2. In the **Main Menu**, click **Advanced** > **Server Automation Scripts**. The Server Automation Scripts List page appears.
3. Search for the **FTOS_DFP_ESign_Download_OnDemand** server automation script and add double-click it. The Edit Server Automation Script page appears.

4. In the **Code** field, search for the code written for downloading the document::

```
var downloadRequest = {
    "envelopeList": [envelopeId]
};
var downloadResponse = JSON.parse(serialize
(dcs.downloadSignedDocument(downloadRequest)));
```

5. Edit the **downloadRequest** as follows:

```
var downloadRequest = {
    "envelopeList": [envelopeId],
    "downloadLog": [true]
};
```

The response will also contain the LogFile in base64 format.

Example

Envelope Log in base64 format

```
{
  "Envelopes": [
    {
      "Name": "signedContract",
      "EnvelopeStatus": "Completed",
      "ExpirationDate": "2019-01-18T09:04:36.16Z",
      "SendDate": "2019-01-16T09:04:36.16Z",
      "MetaData": null,
      "SubEnvelopes": [
        {
          "RecipientEmail": "",
          "FinishedDocuments": {
            "DocumentList": [
              {
                "DocumentId": "6ba9636e-
ef75-425f-8b39-47f428ed7801",
                "DocumentName": "test.pdf",
                "FormFieldValues": null
              }
            ],
            "LogDocId": "505598f5-e138-4336-
8ce8-3c553dcfec1f"
          }
        }
      ]
    }
  ]
}
```

```

        "Recipients": []
      }
    ],
    "EnvelopeId": "684551f2-f5b9-4eb1-a514-
c7eea0be83dd",
    "Files": [
      {
        "FileName": "test.pdf",
        "Data": "base 64 file"
      }
    ],
    "LogFiles": [
      {
        "FileName": "EnvelopeLog_03a5fbfc-480a-
4a6d-a796-a7121e46d794.pdf",
        "Data": "base 64 file"
      }
    ]
  }
],
  "IsSuccess": true,
  "ErrorMessage": null
}

```

Digital Documents Processor

The Digital Documents Processor enables you to leverage intelligent document automation to reduce errors, boost productivity and maximize business outcomes.

Easy to use, this automation processor allows you to automatically generate dynamic, personalized and accurate essential business documents – including customized contracts and agreements, by merging real-time data. You can later use these documents in digital journeys to minimize the paperless flows, reducing workload and completion times, as well as increasing the customer’s satisfaction.

Installation

Innovation Studio comes with the Digital Documents Processor automation processor pre-installed.

Applications

The Digital Documents Processor can facilitate business processes, such as:

- Customer onboarding
- Account opening
- Policy application
- Retail Current Account Onboarding
- Loan applications.

Setting Up a Digital Document

Digital documents require as a prerequisite the creation of a report as well. To do so:

1. Create a file-type attribute within the entity you wish to attach the document.
2. Open the **Analytics** main menu, select **Report** item.
3. The Add report page will open. Fill in the following fields:

Field	Description
Name	Insert an appropriate name for the report.
Display Name	This name will be displayed in the UI.
Scope	Select the scope "Entity".
Type	Select the type "Document".

Field	Description
Entity	Select the name of the entity from where to use the tokens and to where store the actual document.
Output Method	Select "Attach to entity."
Destination Field	Select the name of the attribute created in step 1. <div style="background-color: #f4a460; padding: 10px; text-align: center;">IMPORTANT! It must be "file" type.</div>
Destination Field Name	Insert the name of the file that will be created.
Report Document Type	Select the type of extension the document will be: <ul style="list-style-type: none"> • DOCX • PDF.

4. Click the **Save and reload** button.
5. Click the **Report Item** to insert the configuration for the fields:

Field	Description
Name	Insert a name for the item.
StartDate	Select a start date from when the item is available.
Enddate	Select an end date from when the item is no longer valid.
Report Document	Select the template or create a new one. For details, see "Creating Document Templates" on the next page.
isDefault	True, then the item is the default one for the report.
Report	This field is read-only. It is automatically filled with the name of the report created in the previous table.

6. Click the **Save and reload** button.

Creating Document Templates

Before creating a digital document in Innovation Studio, you should create a template using MS Word or MS Excel.

IMPORTANT! To avoid breaking the generated digital documents, when creating the document templates, make sure that:

- you remove all comments.
- to accept all changes (if any) and stop tracking changes.
- the signature details fit within the page, always center the signature token.

When creating the template, use token fields for the fields and table tokens. The template automatically fills in with data from the DB, as it is going to be attached to a digital document.

NOTE

The digital document's data source is an SQL procedure or Fetch Collection. The SQL procedure has to be defined in advance, while the Fetch Collection is defined after attaching the template to the digital document.

Use token fields

A token field is a text field that includes a block of text (token) that can be easily selected and manipulated.

To use token fields within your report template, include them within curly brackets {} without leading or trailing spaces.

NOTE

The token should have the following format: table.Name.attributeAlias or tableName.attributeName.

The figure below shows an example of a report template.

Use table tokens

In case of a table token, the SQL procedure call must contain two select queries separated by comma:

- The first SELECT should follow this pattern: SELECT 'tokenName' as 'table' to specify that the 'tokenName' token represents a table.
- The second SELECT will return effective data for the table. The document table columns name must be identical to the attributes name returned by query.

Example:

```
select 'views' as 'table',
select entityviewid as 'entityviewid', Data as 'Data'
from EbsTestEntityView
```

Format tables in DOCX and XLSX templates

You have various possibilities to format tables in templates: table size, border size, cell padding, add/delete header and border color. You can also include datasets within the table by using table tokens which you handle via SQL procedures or Fetch Collections.

When creating the document template, format the tables as needed.

To automatically fill-in table rows with data from the DB, add a new row to the table and provide the table token in the following format: {tableName.columnAlias}

Example

How to configure a table in the .docx or .xlsx template:

HEADER col 1	Header col 2	Header col3
{table2.col1}	{table2.col2}	Static text
{table1.column1}	{table1.column1}	{table1.column1}

This row will be replaced by the data returned as per the provided SQL procedure. You will provide both tableName and columnAlias in the SQL procedure.

SQL procedure

```

CREATE PROCEDURE [dbo].[uspTest]
    @Id uniqueidentifier
AS
BEGIN
    --the first data set should return only one row
    select 1 as 'abc'

    select 'table1' as 'table' -tableName
    -- data set for table1
    select name as column1
    ,createdOn as column2
    from ebs.TestTable1

    select 'table2' as 'table' -tableName
    -- data set for table2
    select ContractName as col1
    ,CreatedDate as col2
    from Ebs.Contract
END
    
```

Table from the generated document based on the SQL procedure provided in the example above

HEADER col 1	Header col 2	Header col3
Test1	TestCol2	Static text
Test2	Test2Col2	Static text
Table1 Info		
Table1 Info 2		
Table1 Info 3		

NOTE

- There shall be only one tableName per table row, otherwise, the rows will be duplicated based on the data from the first table provided in the SQL procedure.
- You can add as many tableName tags in a table as long as they are on different rows.

- If the procedure returns no dataset, the row is removed from the table within the template which contains the specific tag.

Table from the document generated with dataset table2 with no rows

HEADER col 1	Header col 2	Header col3
Table1 Info		
Table1 Info 2		
Table1 Info 3		

Insert a barcode in your DOCX template

In order to insert a barcode in your DOCX template, go to Innovation Studio > **Admin > System Parameters** and change the value of the `sys-documentreport-should-read-barcode-from-userfiles-storage` system parameter to:

- 0 - in order to declare in the SQL procedure the absolute path to the barcode image. E.g.:

```
select 1 as Txt1,
'C:\Users\john.doe\source\repos\master-
new\ebscore\EBS.Core.Web.MVC\UploadEBS\document-
report-azure.png' as 'barCode128'
```

- 1 - in order to declare in the SQL procedure the relative path to the barcode image. E.g.:

```
select 1 as Txt1,
'document-report-azure.png' as 'barCode128'
```

Add the {barCode128} token in your template along with the rest of the token fields and upload the template in your Digital Document.

NOTE

Each template must contain only one {barCode128} token.

Show or hide objects in document templates

You can show or hide/remove objects (that is, paragraphs or tables) in the Word/Excel document templates.

To do so, use the following token: {show=var1}, where **var1** is provided in the SQL procedure in the first dataset.

In the SQL procedure, the possible values for **var1** are **1** (show object) and **0** (hide object); whereas, the default value is **1**.

If the **var1** value is not provided within the first dataset of the procedure, the object will be displayed.

To hide an object, in the SQL procedure set the value of **var1** to **0**.

Example

Word document template with the show token:

{Show=var1}

• {table2.col1}	{table2.col2}	
{table1.col1}		

SQL procedure with var 1 set to value 1

```
ALTER PROCEDURE [dbo].[testDocReport]
    @Id uniqueidentifier
AS
BEGIN
    select 'Doc00001' as 'DocumentNo'
    , 'NEW_Doc00001' as 'FtosReportFileName'
    , 1 as 'var1'

    select 'table1' as 'table'
    -- data set for table1
```

```

select 1 as 'col1'
,2 as col2
,3 as col1
from ebs.Test

select 'table2' as 'table'
-- data set for table2
select 2 as 'col1'
,1 as col2
from ebs.Test

END
    
```

The document generated based on the template and SQL procedure provided in the previous examples looks like this:

• <i>Contract1</i>	Test 1	
{table1.col1}		

Creating Digital Documents

In Innovation Studio, you can create three types of digital documents based on the data source used upon document generation, as follows:

- Stored SQL procedures - gathers data from a stored SQL procedure.
- Entities - gathers data from the entity on which the digital document is generated.
- Fetch collections - gathers data as per defined fetch collection.

For information on how to add and configure digital documents, see below the section corresponding to the data source that you want to use:


Defining Digital Documents using Stored SQL Procedures

During this configurations, the document is created based on the template attached while gathering the source data for a SQL procedure, batch of statements grouped as a logical unit and stored in the database with parameters or one at all.

Prerequisites

- You should have created the SQL procedure that you want to use when creating the report document.
- The SQL procedure should control the Date and Numeric formatting (culture specific).

Add a digital document using SQL Procedure

1. Open Innovation Studio in developer mode.
2. Click the Main Menu icon at the top left corner.
3. Click **Fintech Automation > Digital Documents**. The Digital Documents List page appears.
4. At the top-right corner of the page, click the **Insert** icon (). The Add Digital Document page appears.
5. Enter a **Name** for the digital document.
6. By default, the selected **Data Source Type** is **Stored SQL Procedure**. Leave it as is.
7. In the **Stored procedure** field, provide the name of the SQL procedure following this convention: *procedure_name_as_stored_in_DB @Id @EntityName @UserId*
8. In the **Template** field, upload the document template to be used . The word document will be uploaded into the system and linked to the current digital document.

The screenshot shows a configuration form titled "ADD DIGITAL DOCUMENT". It contains the following fields:

- Name:** ContractStoredProc
- Data Source Type:** SQL Stored Procedure
- Stored procedure:** procedure_name_as_stored_in_DB @Id @EntityName @UserId
- Date and Numeric Formatting:** (empty field)
- Template:** Terms and Conditions.docx

The uploaded document template is saved into the "DocumentReportTemplates" folder which is a sub-folder of "UploadEbs".

9. Click the **Save and Close** button at the top right corner of the page.

The following parameters of the stored SQL procedure are automatically mapped to specific values, as described in the table below.

Parameter	Value mapped to
@Id	The record ID of the entity item that has the Report linked to the Report Document.
@EntityName	The name of the entity that has the Report linked to the Report Document;
@UserId	The ID of the user that runs the Report linked to the Report Document.

Defining Digital Documents using Entity Data

During this configurations, the document is created based on the template attached while gathering the source data for an entity and its attributes.

1. Open Innovation Studio.
2. Click the Main Menu icon at the top left corner.
3. Click **Fintech Automation > Digital Documents**. The Digital Documents List page appears.

4. At the top-right corner of the page, click the **Insert** icon. The Add Digital Document page appears.
5. Enter a **Name** for the digital document.
6. From the **Data Source Type** field, select **Entity**.
7. Optionally, from the **Date and Numeric Formatting** field, select the culture that will apply to dates and numbers.
8. In the **Template** field, upload the document template to be used . The word document will be uploaded into the system and linked to the current digital document.

The uploaded document template is saved into the "DocumentReportTemplates" folder which is a sub-folder of "UploadEbs".

9. Click the Save and Close button () at the top right corner of the page.

Defining Digital Documents using Fetch Data

During this configurations, the document is created based on the template attached while gathering the source data using Fetch Designer, a Innovation Studio feature which pulls data in a no-code manner. It can join entities while adding conditions with the operands "AND" and "OR".

Add a Digital Document:

1. Open FintechOS Studio in developer mode.
2. Click the Main Menu icon at the top left corner.
3. Click **Fintech Automation > Digital Documents**. The Digital Documents List page appears.
4. At the top-right corner of the page, click the **Insert** icon. The Add Digital Document page appears.
5. Enter a **Name** for the digital document.
6. From the **Data Source Type** field, select **Fetch collection**.
7. Optionally, from the **Date and Numeric Formatting** field, select the culture that will apply to dates and numbers.
8. In the **Template** field, upload the template to be used . The MS Word or Excel document will be uploaded into the system and linked to the current digital document.

ADD DIGITAL DOCUMENT

DIGITAL DOCUMENT

Name

Data Source Type

Stored procedure

Date and Numeric Formatting

Template

or Drop file here

FETCH COLLECTION

The uploaded document template is saved into the "DocumentReportTemplates" folder which is a sub-folder of "UploadEbs".

9. Click the Save and Reload button at the top right corner of the page. The Edit Digital Document page appears

IMPORTANT! You have to define at least one fetch; otherwise the digital document cannot be generated.

The digital document will get its data from the entity record that is opened when generating the document.

Define Fetches

Prerequisites

- Make sure that there are at least two entities in the system.
- Create relationships between the entities on which you do the fetch.
- Add custom attributes to each entity for which you do the fetch; the attributes that you will use when defining the fetch.

For more information on entities, attributes and relationships, see the *Innovation Studio User Guide*, section *Data Model Explorer*.

To define a fetch, follow these steps:

1. In the Edit Digital Document page, scroll-down to the Fetch Collection section and click the **Insert** button. The Add Fetch Definition page appears.
2. Enter a **Name** for the digital document.
3. In the **Fetch definition** field, fetch the data which will be used for generating the report.

To retrieve the value of a document parameter (when the document is generated, use the `getParameter` function.

Example: Retrieve the value of document parameter param1

```

var p1 = getParameter("param1");
return {
  "entity": {
    "alias": "base",
    "name": "entity",
    "attributelist": [
      {
        "name": "displayName",
        "alias": "",
        "attributeType": 3
      },
      {
        "name": "name",
        "alias": "",
        "attributeType": 3
      }
    ]
  },
  "where": {
    "type": "and",
    "conditionlist": [
      {
        "first":
"base.tableName",
        "type": "equals",
        "second": "val(" + p1+
")"
      }
    ]
  }
}

```

You can also define a fetch by using the Fetch Designer (clicking the **Show Fetch Designer** button). For information on how to use the Fetch Designer, see the [Innovation Studio User Guide](#), section [Fetching Entity Data](#).

4. The fetch definition is saved into the system and it displays in the Fetch Collection section.

The following fetch parameters are automatically mapped to specific values, as described in the table below.

Parameter	Value mapped to
Id	The record ID of the entity item that has the Report linked to the Report Document.
EntityName	The name of the entity that has the Report linked to the Report Document;
UserId	The ID of the user that runs the Report linked to the Report Document.

Set the fetch collection execution order

If you have multiple fetches defined in the Fetch Collection, you can set their execution order by drag and dropping rows in the Fetch Collection section, whereas the first fetch has the order index 1, that is, will be executed first.

The placeholders in the document template (word document) will be replaced following these rules:

1. The first fetch (with order index 1), will always return one record which is used to define the Headers.
 - If the fetch attribute does not have an alias, then the placeholder name is based on entity alias and fetch attribute name. The format is `{entityalias_attributename}`;
 - If the fetch attribute has an alias then the placeholder is the alias, `{alias}`;

Using the example above: for the “name” attribute the placeholder should be `{base_name}` and for “entityid” attribute the placeholder should be `{exampleAlias}`.
2. Fetches other than the first one (with order index higher than 1) will be used to populate tables; therefore, they might return many records.
 - If the fetch attribute does not have an alias then the placeholder name is based on fetch name, entity alias and fetch attribute name. The

format is {fetchname.entityalias_attributename};

- If the fetch attribute has an alias then the placeholder name is based on fetch name and attribute alias, {fetchname.alias};

Attaching a Report to the Entity pointing to the Document

To create a document successfully, the user needs to create a report type "document" where a report item will be added.

1. Open Innovation Studio.
2. Click the Main Menu icon at the top left corner.
3. Click **Evolutive Data Core > Data Model Explorer**. The Business Entities List page appears.
4. Search for the entity the user wishes to attach the document to. Double-click the entity to open its configurations.
5. Navigate to the **Data Model** where the attributes are.
6. At the top of the grid, click **Insert**.
7. Fill in the fields for the attribute where the documents will be stored.

Field	Data type	Description
Name	Text	Insert a suggestive name.
Attribute type	Option set	Select from the from-down list, the File type.
Display name	Text	Insert a suggestive name.
Description	Text Area	Insert a suggestive description.

Field	Data type	Description
Tooltip	Text Area	Insert a message to be rendered on the field when hovering over it.
Table column name	Text	This field is automatically filled in.
Restrict files number	Boolean	If true, then the number of files to be stored here is restricted to a specific number, e.g. 10578 documents.
Maximum number of files	Whole number	If the boolean from above is true, insert the number here. If false, ignore this field.
Required Level	Option set	Select where this attribute is required in a data form: <ul style="list-style-type: none"> • none • recommended • required.
Is readonly	Boolean	If true the field cannot be modified.
Is securable	Boolean	If true, only the specific security roles are allowed to see it. For details, see Creating Security Roles .

ADD DIGITAL DOCUMENT

DIGITAL DOCUMENT

Name

Data Source Type ✖ ✎

Stored procedure

Date and Numeric Formatting ↓ ✎

Template ✖
 or Drop file here

FETCH COLLECTION

8. Click **Save and Close**.
9. Navigate to **Analytics**.
10. Select **Reports**. The Reports List page appears.
11. Click **Insert**. The Add Report page appears.

Field	Data type	Description
Name	Text	Insert a suggestive name.
Display name	Text	Insert a suggestive name.
Scope	Option set	Select Entity .
Type	Option set	Select Document .
Entity	Lookup	Select the entity where the user inserted the attribute - type file.
Output method	Option set	Select attach to entity.
Destination field	Text	Insert the name of the attribute- type file- created earlier.
Destination File Name	Text	It is the name of the folder in the server where the document will be downloaded.

Field	Data type	Description
Report Document Type	Option set	Select PDF .

- Click **Save and reload**. The Edit report page displays.
- To match the documents processor with the report, navigate to the **Report items** grid. Click **Insert** to add an item.
- The page Add Report item is displayed. Fill in the following fields:

Field	Data type	Description
Name	Text	Insert a suggestive name.
Start date	Date	Select the date when the report item will start.
End date	Date	Select the date when the report item will end.
Report document	Lookup	Select from the list the document template created in this automation processor.
Is default	Boolean	If the boolean is true, then the document template will be the default one for this report.
Report	Lookup	It is automatically filled in with the name from the report.

- Click **Save and close**.

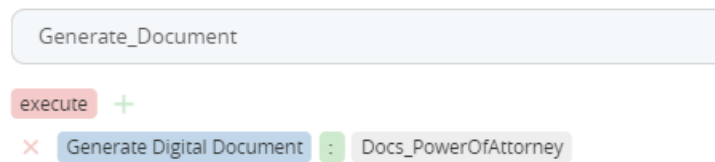
Using the Document in the Form Driven Flow

Calling the generation of a document is possible by:

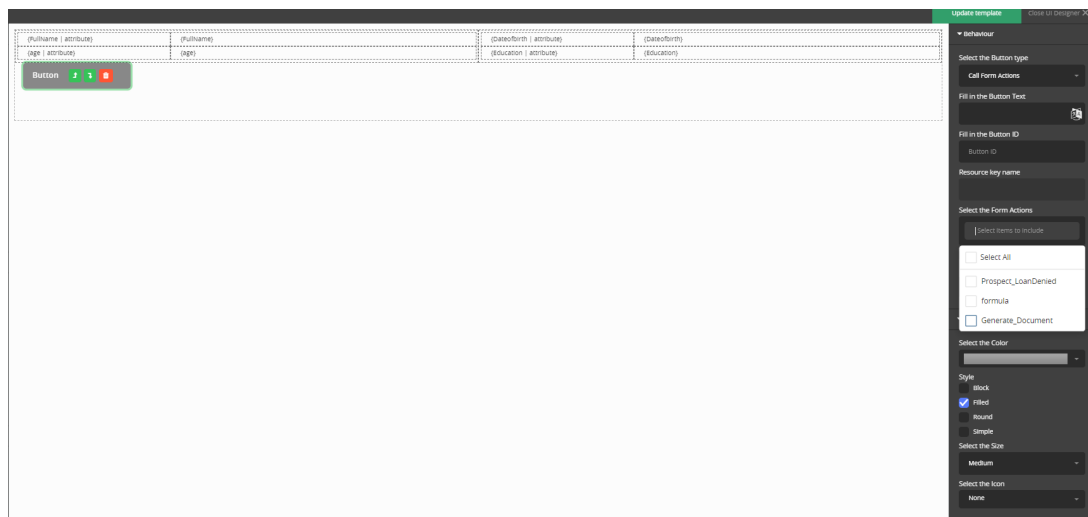
- Open Innovation Studio.
- Click the Main Menu icon at the top left corner.
- Click **Digital Journey > Form driven Flow**. The Form Driven Flow List page appears.

4. Search for the form driven flow where the user wishes to use the document.
5. Navigate to the **Actions** tab. Click on it.
6. In the grid, click the **Insert** button to add a new action.

Form Action



7. Click the **Save** button on the right-hand corner of the page.
8. Navigate to the step where the action will take place. Select the step from the **Steps** tab.
9. Navigate to the UI tab of a particular step. Open the UI designer.
10. Add a button in the layout. Its configurations open in the right-hand part of the screen.



11. Select the type **call form action** and in the **Select from action field** tick the action created earlier which generates the document.
12. Click the **Apply changes**, then click **Update template**, and lastly **Save and close**.

Automatically Generate Customer Contracts

In this example, we set up a button on a customer edit form that generates a services agreement automatically populated with the customer's name.

1 Prepare the contract template.

Create a contract template in Microsoft Word. Enclose between curly brackets any entity attribute names that must be populated automatically. For more details, see ["Creating Document Templates" on page 256](#).

2 Set up a digital document based on the contract template.

IMPORTANT!

For the entity you wish to attach the document, create a file-type attribute where to store the document itself.

Create a new entity-based digital document using the template created at Step 1. For details, see .

3 Attach a report to the entity based on the digital document.

1. Make sure that the target entity (in this case *Investor*) has a file attribute defined in which to store the contract (in this case *contract*).
2. In Innovation Studio, go to **Reporting & Analytics > Reports** and create a new report attached to the contract file storage attribute defined in the target entity.
 - Name – Unique name used to identify the report in the system.
 - Display Name – How the report name will be displayed in the user interface.

- Scope – Entity.
 - Type – Document.
 - Entity – Name of the entity to which you wish to attach the report (in this case *Investor*).
 - Output Method – Attach to entity.
 - Destination Field – Name of the entity attribute that will store the report (in this case *contract*). The attribute must be of file type.
 - Destination File Name – File name under which the report will be saved.
 - Report Document Type – File type under which the report will be saved.
3. Click the **Save and Reload** button at the top right corner of the page.
 4. In the Report Items section, insert a new entry for the digital document you created at Step 2.
 - Name – Enter a custom name for the report item, or leave the default name in place.
 - Start Date and End Date – Upon the report generation, it will gather data within the specified time interval (between the start date and the end date).
 - Report Document – Select the digital document created at Step 2.
 - IsDefault – Select the checkbox so that the report item is used when generating the report.
 5. Click the **Save and Close** button at the top right corner of the page.

4 Create a button to generate the customer contract.

1. In Innovation Studio, open the customer's entity form or form driven flow where you want to generate the contract.
2. In the UI designer, add a button to generate the contract.

3. In the **Advanced** tab, in the **After events** section, edit the button click event to trigger the report generation.

```
/* Click event for the generateContract button */
$('#generateContract').on('click', function (event) {
    ebs.callReportByName('wealthManagementContract', formData.id);
});
```

4. Click the **Save and Close** button at the top right corner of the page.

5 How to generate a customer contract from the user interface

1. Open FintechOS Portal.
2. Open the entity form or form driven flow where the button to generate the customer contract is located.
3. Make sure the data required to generate the contract is populated and saved (in this case the *name* field).
4. Click the button to generate the customer contract.
5. The customer contract will be generated and saved in the designated file attribute.
6. You can click the file name to download the file locally.

Campaign Management

Campaign management is the process of planning, organizing, executing, and tracking different types of campaigns that serve multiple purposes: marketing, sales, administrative, and so on.

Throughout the buying journey, customers bounce between different digital channels (mobile devices, laptops, etc.) so marketers need to ensure that customers receive the same seamless experience no matter which channel they use. Moreover, the content needs to be personalized based on each customer needs.

Use the below FintechOS components to easily create, manage, and personalize your campaign.

Omnichannel Campaigns

Fintech marketing campaigns are essential for customer engagement, loyalty, and most importantly they can inform and educate the customer when it comes to lending and loan processes, financial application updates, or any other banking or insurance processes.

These campaign types can be primarily directed towards bank clients through regular campaigns such as fall/ winter season, holiday season campaigns, or internal campaigns that concern financial institutions employees.

In this day and age, customers bounce between channels (mobile devices, laptops, etc.) throughout the buying journey, so marketers need to ensure that no matter which channel the customers use, the message is seamless - customers receive the same experience and messaging through each and every channel.

Omnichannel campaigns has the customer at the core to ensure a unified experience at every touch point in the buying journey, fostering an effortless buying experience for customers.

The Omnichannel Campaigns automation processor empowers you with the ability to create effective and user-tailored ways of interacting with the customer. Using this automation processor, you can automate completely personalized campaigns, populate unique emails for each individual, ensuring personalized communication with your customers based on their needs.

This is done in a cost-effective manner, and through the modular design of the FintechOS High Productivity Financial Infrastructure (HPFI), it ensures both encapsulation and integration. You can easily change how interaction with a certain audience is done via campaigns, seasons, and marketing tools.

Omnichannel Campaigns Features

- Uniquely identify the members of imported audience lists
- Anti-spam functionality, with options to skip or delay message on bank days or after previous communication
- Staged campaign execution, with distribution variations on channel, content template and A/B Control Group
- Redirecting messaging to campaign controlling group
- Advanced recurrence settings per stage
- Previewing scheduled activities and exceptions (exceeded length, no data available for token, etc.)
- Approval workflow to manage campaign authoring
- Previewing and exporting execution plan and campaign activities, logs and A/B variation counters for each run

Applications

The Omnichannel Campaigns module can aid business processes such as:

- Loan applications
- Quote applications
- Mortgage processing.

Installing Omnichannel Campaigns

Follow the steps described below to perform an automatic installation of the Omnichannel Campaigns. This is a process of running `install_SysPack.bat` files on your environment.

IMPORTANT!

You must run the script on the machine where Innovation Studio is installed. Make sure you have access rights to Studio's database.

Dependencies

In order to install Omnichannel Campaigns, the following needs to be installed first:

- **Innovation Studio** minimum version v22.1
- **Standard SysPack** minimum version v22.1.0001
- **FTOS.Foundation - Project**

Pre-Installation Checklist

The SysPack has a unique constrain on the following entities: `FTOS_MKT_AudienceSegments`, `FTOS_MKT_Audience`.

If you have already moved data using the **Configuration Data Deployment Package** menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

1. Disable the constraints that you have created on your environment, allowing the system to create the new one after Omnichannel Campaigns is imported.

2. Use the new **Configuration Data Definitions** imported with the Omnichannel Campaigns file when you export the data.

Installation Steps

Make sure you have the **SysPacks v.22.1.1000** installed on your system. To do so:

1. Using a web browser, log in to your [FintechOS Community](#) account.
2. Select the **Release Hub**.
3. Open the **FintechOS 22.S** release.
4. Open the **HPFI** folder.
5. Download the **SySDigitalSolutionPackages v22.1.0001.zip** archive.
6. Unzip the archive and follow the instructions from the [SysPacks Installation](#) page.

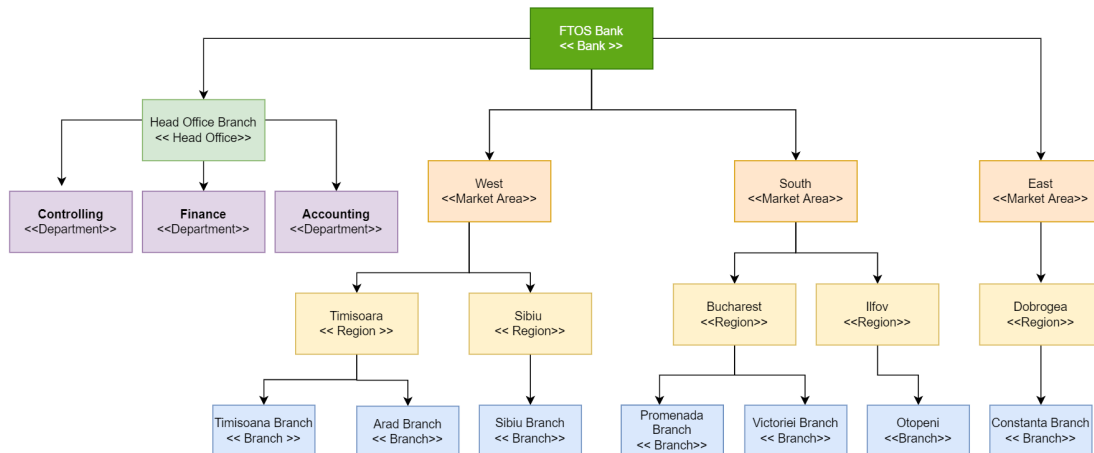
Creating the Organizational Structure

Having an organizational structure is necessary in order to achieve the organization's objectives as it outlines not only each employee's role and how it fits within the system, but also a bank's internal organizational structure.

To create the organizational structure, log into the FintechOS Portal and follow the steps from the below pages:

- [Adding System User Information](#)
- [Adding Business Unit Branches](#)

The example below helps you understand a bank's internal organizational structure:



Adding System User Information

Follow the below steps when adding system user information:

1. Go to the FintechOS Portal.
2. Open the **General** menu in the top left corner.
3. Select the **System User Information** menu option. The **System User Information List** page is displayed.
4. Click the **Insert** button in the top right corner. The **Add System User Information** page is displayed.
5. Fill in the following fields:

AUTOMATION BLOCKS USER GUIDE

ADD SYSTEM USER INFORMATION

SYSTEM USER INFORMATION

Branch	Hub Sector 4  
Function	Relationship Manager
Manager	host  
Managers Path	DocTest
Name	DocTest

Field	Type	Required	Description
Branch	Lookup	No	The branch that the user belongs to. Select from the list or create a new record. For additional details see the Adding Business Unit Branches section.
Function	Text	No	The user's function.
Manager	Lookup	No	The manager name. Select from the list or create a new user based on the steps from the Adding Users page.
Manager Path	Text	No	The manger path.
Name	Text	No	The user's name.

6. Click **Save and close** to return to the **System User Information List** page.

Adding Business Unit Branches

Business unit branches represents a great way to set up a bank internal organizational structure. Follow these steps to add new business unit branches:

1. Go to the FintechOS Portal.
2. Open the **General** menu in the top left corner.
3. Select the **Business Unit Branch** menu option. The **Business Unit Branches List** page is displayed.
4. Click the **Insert** button in the top right corner. The **Add Business Unit Branch** page opens.
5. Fill in the following fields:

EDIT BUSINESS UNIT BRANCH

BUSINESS UNIT BRANCH

Name	BranchLast
Unit Type	FTOS_ACC_UnitType_last ↓ ✎
Parent Unit Type	FTOS_ACC_UnitType_1 ↓ ✎
Master Business Unit Branch	Bucuresti ↓ ✎
Referenced Business Unit	BranchLast ↓ ✎

Field	Type	Description
Name	Text	The name of the branch.
Unit Type	Lookup	The unit type of the record. Lookup to FTOS_ACC_UnitType.
Parent Unit Type	Lookup	The parent type of the current record. Lookup to FTOS_ACC_UnitType.

Field	Type	Description
Master Business Unit Branch	Lookup	Filtered lookup, displaying only business unit branches with Parent unit type.
Referenced Business Unit	Lookup	Business unit from the business unit system entity. It is automatically filled in.

7. Click **Save and reload**. The **Business Lines** section is displayed.
8. Click the **Insert existing** button to add a business line or create a new one. The following options are available:
 - Corporate
 - Individuals
 - Insurance
 - WM
9. Click **Save and close** to return to the **Business Unit Branches List** page.

Omnichannel Campaigns Management

Campaigns typically aim to reach consumers in a variety of ways to promote a specific purpose such as reminder for a bank transfer, policy termination and various client notifications about product releases as well.

This chapter covers the following topics:

Managing Campaign Types

Campaign types are a quick and convenient way of categorizing campaign data. For example, campaigns for the fall/ winter season, holiday season campaigns, and so on. Each campaign can be assigned a certain type and subtype. Campaign types can contain multiple campaign subtypes.

Each campaign can be assigned a certain type (fall/ winter season, Easter, etc.) and subtype (mail, SMS campaign, etc.). Campaign types might contain multiple campaign subtypes.

Adding Campaign Types and Subtypes

To add a campaign type, follow these steps:

1. Expand the main menu icon at the top left corner.
2. In the main menu, navigate to the **Omnichannel Campaigns** menu and select **Campaigns Types**. The **Campaigns Types List** page is displayed.
3. Click the **Insert** button at the top right corner of the page. The **Add Campaign Type** page is displayed.
4. Fill in the following fields:

Field	Required	Type	Description
Code	Yes	Text	The code of the campaign.
Name	Yes	Text	The name of the campaign. For example: fall/ winter season campaign.

AUTOMATION BLOCKS USER GUIDE

5. To add subtypes to the campaign type, click the **Save and reload** button at the top right corner of the page. The **Campaign Subtype** section is displayed.
6. Click the **Insert** button. The **Add Campaign Subtype** page is displayed.
7. Fill in the following fields:

Field	Required	Type	Description
Code	Yes	Text	The code of the campaign subtype.
Name	Yes	Text	The name of the campaign subtype.
Campaign Type	Yes	Lookup	Select from the existing campaign types or create a new one.

Field	Required	Type	Description
Priority	No	Option Set	<p>To differentiate and decide between the importance of different campaign subtypes, select the campaign priority. The following options are available:</p> <ul style="list-style-type: none"> • High • Medium • Low • Very Low

8. Click **Save and Reload**. The **Edit Campaign Subtype** page is displayed.
9. Click **Save and Close** once the campaign type and subtype have been created. Follow this procedure to add as many campaign subtypes as you need.

Editing Campaign Types

To edit a campaign type, in the **Campaign Types List** page, double-click on a record to open it. The **Edit Campaign Type page** is displayed.

On this page, the campaign name and description can be changed but also campaign subtypes, if any, can be added, edited or deleted. Click the **Save and Close** button at the top right corner to save the changes.

Deleting Campaign Types

To delete a campaign type, in the **Campaign Types List** page, select the desired record and click the **Delete** button at the top right corner of the page. A confirmation dialog appears. Click **Yes** to delete the selected record.

Managing Status Reasons

Campaigns have different purposes, for example loan sales campaigns, savings products sales campaigns, customer activation campaigns, administrative campaigns, and so on. Thus, for generated internal campaigns activities, dedicated reasons must be set when closing campaign activities with an **In progress** or **Cancelled** reason. With this feature a user can define status reasons and status reason templates to cover the need for any type of internal campaigns. Users can set one or multiple status reasons that later can be added to a reason template.

There are two steps that need to be followed when managing status reasons: create the status reasons list and afterwards, set up the reasons template where the status reasons can be included. To create dedicated status reasons and also status template reasons, see the below pages:

- [Defining Status Reasons](#)
- [Defining Status Reasons Template](#)

Defining Status Reasons

Here, a list of status reasons is defined. These reasons are later used when closing an internal campaign activity with an **In Progress** or **Cancelled** status that can be later added to a [status reasons template](#). To add a new status reason, in the Innovation Studio main menu, go to **Automation Blocks > Omnichannel Campaigns > Status Reasons**. The **Status Reasons List** page appears. Select a status from the list or create a new one.

Click the **Insert** button from the right upper corner. The **Add Status Reason** page is displayed. Fill in the following fields:

AUTOMATION BLOCKS USER GUIDE

ADD STATUS REASON

STATUS REASON

Status Reason Code

DocTest

Status Reason

Client didnt answer

Field	Required	Type	Description
Status Reason Code	No	Text	A unique identifier associated to corresponding the status reason value.

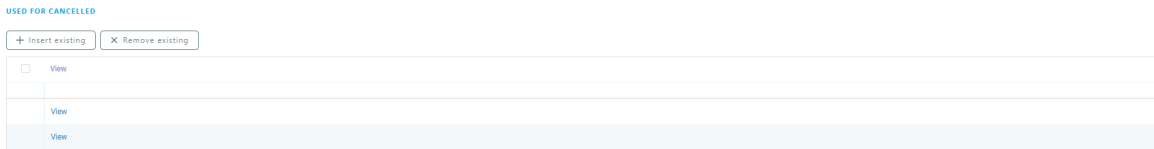
Field	Required	Type	Description
Status Reason	Yes	Option Set	<p>The business status reason set by the person responsible for the campaign activity. This status is set when the campaign status is In Progress or Cancelled. The predefined status reasons are:</p> <ul style="list-style-type: none"> • Assigned • System Aborted • Unassigned • Converted • Completed <div style="background-color: #e1eef6; padding: 10px; border: 1px solid #d9e1f2;"> <p>NOTE Status reasons can be selected from the existing option set values or a new one can be added. To add a new status reason item to the option set, click the edit button. For more details on inserting option set items, see the Adding Option Set Attributes</p> </div>

Field	Required	Type	Description
			page.

Click the **Save and Reload** button. The **Used For Cancelled**, **Used For In Progress**, and **Campaign Activities** sections are displayed.

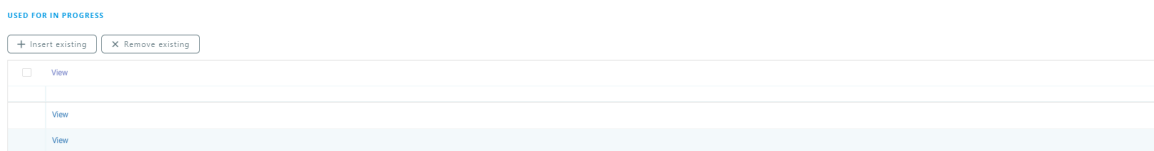
Used for Cancelled

This section displays the reasons templates that have a **Cancelled** status reason, or where other templates using these status reasons can be added.



Used for In Progress

This section displays the reasons templates that have an **In Progress** status reason, or where other templates using these status reasons can be added.



Campaign Activity

This section displays a list of internal campaign activities set with an **In Progress** or **Cancelled** status reason that used a specific status reason. For example, cancelled campaigns with the **Client Refused Offer** status reasons.

AUTOMATION BLOCKS USER GUIDE

CAMPAIGN ACTIVITIES

Export Refresh

Campaign	Activity Type	Activity Subtype	Activity status	Account Name	Account PIN	Fiscal Registratio...	Responsible User	Responsible Bra...	Campaign Priority	Activity date	End Date	Detailed Status	Status Reason	Closed On
DistributionByLo...	Sales		In Progress	Maybell		SMTP*****		BCR01	High	10/01/2022 12:59	31/01/2022	In progress	Client didn't ans...	
DistributionByLo...	Sales		In Progress	Maybell		SMTP*****		BCR01	High	10/01/2022 14:33	31/01/2022	In progress	Client didn't ans...	

5 10 20

Defining Status Reasons Template

Here, a status reasons template is defined using the status reasons created in the **Add Status Reason** page. To add a new status reason template, in the Innovation Studio main menu, go to **Automation Blocks > Omnichannel Campaigns > Reasons Templates**. The **Status Reasons List** page appears. Select a template from the list or create a new one.

REASONS TEMPLATES LIST

Template Name	Activity Type	View
Q	Q	
Credit Line Campaing	Sales	View
Digital campaign	Sales	View
MyAdministrative1	Administrative	View
MyAdministrative2	Administrative	View
ReasonsSales	Sales	View

5 10 20 1 2

Fill in the following fields:

Field	Required	Type	Description
Activity Type	Yes	Option Set	The campaign activity type. The following options are available: <ul style="list-style-type: none"> Sales Administrative
Stage	No	Lookup	The campaign stage. Select from the available records or add a new one. For details on adding campaign stages, see the Scheduling the Campaign in Stages page.

Field	Required	Type	Description
Template Name	Yes	Text	The template name.

Click the **Insert** button from the right upper corner. The **Add Status Reasons Templates** page is displayed. Fill in the following fields:

ADD REASONS TEMPLATE

REASONS TEMPLATE

Activity Type - ✎

Stage ↓ ✎

Template Name ✎

Click the **Save and Reload** button. The **Used For Cancelled**, **Used For In Progress**, and **Campaign Stages** sections are displayed.

NOTE
 Created templates are available for selection when defining a stage for internal campaigns, filtered by activity type.

Used for Cancelled Status Reason Template

In this section the user can add **Cancelled** status reasons applicable for the current template.

CANCELLED STATUS REASONS

View
View
View
View
View

Used for In Progress Status Reason Template

In this section the user can add **In Progress** status reasons for the current template.

AUTOMATION BLOCKS USER GUIDE

IN PROGRESS STATUS REASONS

[+ Insert existing](#) [X Remove existing](#)

<input type="checkbox"/>	View
<input type="checkbox"/>	View
<input type="checkbox"/>	View

Campaign Stages

This section displays the stages that have a certain reasons template set. The reasons template is selected at stage level and the reasons from that template are available when the campaign activities generated from that stage are set to **In Progress** or **Cancelled**.

CAMPAIGN STAGES

[Export](#) [Refresh](#)

<input type="checkbox"/>	Name	Start Date	End Date	Campaign
<input type="checkbox"/>	Q stage	Q	Q	Q
<input type="checkbox"/>	Stage 1	28/10/2021	29/10/2021	testc399_01and_2021-10-28-0449
<input type="checkbox"/>	stage1	01/11/2021	01/11/2021	ext camp
<input type="checkbox"/>	stage run for x days	01/11/2021	02/11/2021	testc399
<input type="checkbox"/>	test stage 1	15/11/2021	15/11/2021	test-c3198_one_instance
<input type="checkbox"/>	stage 1	25/11/2021	25/11/2021	Silviu3_000001

5 10 20 1 2 3 4 5 6

Managing Seasonal Campaigns

Seasonal campaigns enables you to create product bundling campaigns that resonate with the sentiments your customers have around that time of the year.

The seasonal campaigns provides you with the following key advantages:

- Increase brand awareness among customers
- Enhance chances to convert one-time customers into loyal customers
- Boost profit in an otherwise quiet period.

FintechOS enables you to go a step further, split your seasonal campaign into various instances. This is particularly useful if you want to offer different levels of promotional discounts at different times during the seasonal campaign (e.g. offer higher discounts the last 2 days prior the winter campaign ends).

Add seasons

To add a season, follow these steps:

1. Click the main menu icon at the top left corner.
2. In the main menu, click **Business Automation > Omnichannel Campaigns > Seasons**.
The **Seasons List** page appears.
3. Click the **Insert** button at the top right corner of the page. The **Add Season** page appears.
4. Type the **Name** of the campaign type (e.g. Winter).
5. Type in the **Description** for the season.

Year	Season	Start Date	End Date
2,021	Spring	01/03/2021	31/05/2021

6. If you want to split your seasonal campaign into instances, click the **Save and reload** button at the top right corner of the page. The **Edit Season** page appears. To add an instance:

Year	2,021	Name	Spring 2021
Start Date	01/03/2021	End Date	31/05/2021

1. In the Instances section, click the **Insert** button. The **Add Season Instance** page appears.
2. Type a **Year** and **Name** . Select the season instance **Start Date** and **End Date**.

3. Click the **Save and Close** button at the top right corner to save the campaign subtype. The **Name** of the instance is automatically set as the concatenation of the season name and the instance year.

Follow this procedure to add as many season instances as you need.

7. Click the **Save and Close** button at the top right corner to save the season settings.

Edit seasons

To edit a season, in the Seasons List page, double-click the record that you want to edit. The Edit Season page appears.

You can edit the name, description and also add, edit or delete season instances (if any), then click the **Save and Close** button at the top right corner to save the changes.

Delete seasons

To delete a campaign type, in the Seasons List page, select the desired record and click the **Delete** button at the top right corner of the page. A confirmation dialog appears. Click **Yes** to delete the selected record.

Managing Marketing Team Members

Storing information about the marketing team members is very easy. This feature is used to record the person responsible for sending specific messages and keeps a log of the number of messages sent. In addition to that, personal information like name, email and phone are also stored along with an associated user to allow an easy way of contacting that person and to promote ownership.

Add marketing team members

To create your marketing team responsible with the customer communication and omnichannel campaigns, follow these steps:

1. Click the main menu icon at the top left corner.
2. In the main menu, click **Business Automation > Omnichannel Campaigns > Marketing Team Members**. The **Marketing Team Members List** page appears.
3. Click the **Insert** button at the top right corner of the page. The **Add Marketing Team Member** page appears.

4. Set the following fields:

Field	Description
Account	An account that already exists in the database.
Attached User	The user attached to the account.
First Name	The first name associated with the account.
Last Name	The last name associated with the account.
Email	The email associated with account.
Phone	The phone number associated with the account.
Team Member Function	Select a function for that team member: <ul style="list-style-type: none"> • manager • account manager.

5. Click the **Save and Close** button at the top right corner to save the team member. Repeat the procedure above to add as many members as you need for your marketing team.

Edit marketing team member details

To edit a marketing team member's details, in the Marketing Team Members List page, double-click the record that you want to edit. The **Edit Marketing Team Member** page appears.

Make the desired changes and click the **Save and Close** button at the top right corner to save the changes.

Delete marketing team member

To delete a marketing team member from the list, in the Marketing Team Members List page, select the desired record and click the **Delete** button at the top right corner of the page. A confirmation dialog appears. Click **Yes** to delete the selected record.

Creating Campaigns

The following chapter of this guide is meant to explain the steps taken when a campaign is created.

IMPORTANT!

Prior to creating campaigns, you need to create personalized content and define audiences using the Hyper-Personalization Automation processor; otherwise, you won't be able to create campaigns. For information on how to create personalized and audiences, see the [Hyper-Personalization Automation Guide](#).

To begin creating the campaign, log into the Innovation Studio and expand the main menu icon at the top left corner.

In the main menu, navigate to **Automation Blocks > Omnichannel Campaigns > Campaigns**. The **Campaigns List** page appears. Follow the steps from the below pages:

Setting up a Campaign

To set up a new campaign, click the **Insert** button at the top right corner of the page. The **Add Campaign** page appears by default on the **Setup** tab. Fill in the following fields:

AUTOMATION BLOCKS USER GUIDE

ADD CAMPAIGN

Name

Start Date End Date

Campaign Type Campaign Subtype

Campaign Priority Campaign Identifier

Total Number of Days No of days since start Remaining Days

Description

Created by user Created On

Field	Required	Type	Description
Name	Yes	Text	The name of the campaign that uniquely identifies it.
Start Date	Yes	Date	The date when the campaign starts. This field is mandatory.
End Date	Yes	Date	The date when the campaign ends. This field is mandatory.
Campaign Type	Yes	Lookup	The campaign type. It allows financial institutions to better organize between their campaigns. For example: Easter campaign, Summer campaign, or even a campaign that offers a special interest rate for a limited time period. For additional details, see the Managing Campaign Types page.

Field	Required	Type	Description
Campaign Subtype	No	Lookup	The campaign subtype. It allows financial institutions to categorize the campaign types. For example, bank customers can be notified about a current campaign via email, telephone, text messages, and so on. For additional details, see the Managing Campaign Types page.
Campaign Priority			The campaign's priority. The following options are available: <ul style="list-style-type: none"> • High • Medium • Low • Very Low
Campaign Identifier	No	Text	Specific to each customer, this is an unique identifier of the campaign.
Total Number of days	No	Numeric	Insert the number of days that the campaign will be taking place.
No days since start	No	Numeric	The number of days since the campaign has started.

Field	Required	Type	Description
Remaining Days	No	Numeric	The number of remaining days, the campaign has until it ends.
Description	No	Text	The purpose of the campaign, how it works, intended audience, etc.
Created by user	No	Text	This field is read-only. It displays the name of the user once saved.
Created On	No	Text	This field is read-only. It displays the date once saved.

Click the **Save and reload** button. The **Edit Campaign** page is displayed. To navigate between sections, click on the bullets from the bullet list which renders the section tabs.

Defining the Campaign Content

Click the **Content** tab. In this section, the way that the audience is contacted is defined.

The screenshot shows the 'EDIT CAMPAIGN' interface. At the top, there are several configuration options:

- Template:** A dropdown menu with 'content@intam' selected.
- Type of delay for holidays:** A dropdown menu with 'Send anyway' selected.
- Type of delay for weekend:** A dropdown menu with 'Send anyway' selected.
- Exclude From AntiSpam:** A checkbox that is checked.
- Max Delay Days No:** An input field.
- Create Activities On Activity Date:** A checkbox that is checked.

 Below these are 'ANTISPAM SETTINGS' with buttons for '+ Insert', 'X Delete', 'Export', and 'Refresh'. At the bottom, there is a table with the following data:

Communication Channel	Source Campaign	Delay Days No
Q	Q	
SMS	TestTeo090222	7
email	DocTest	7

The content is selected by choosing a template. Content templates are essential to the content creation process as they provide the means to interact with customers in

a meaningful manner. Special options exist for anti-spam, namely the possibility to select delays for weekends and special holidays. Thus, a delay for the communication is added if a message is scheduled to be sent on a weekend or on a holiday.

In addition, there is the option to exempt the communication from anti-spam rules. This is useful for critical messages that need to reach the audience right away.

The table below describes the settings for defining the campaign content:

Field	Type	Description
Template	Lookup	Select the personalized content to use from the list of available templates. For details, see the Managing Personalized Content page.
Type of delay for holidays	Option Set	The type of delay to be added to the communication date, if the communication date occurs on a holiday. The following values are available: <ul style="list-style-type: none"> • Do not send • Send anyway • Send After
Type of delay for weekend	Option Set	The type of delay to be added to the communication date, if the communication date occurs during the weekend. The following values are available: <ul style="list-style-type: none"> • Do not send • Send anyway • Send After
Exclude From AntiSpam	Bool	If true , it excludes the communication from the anti-spam rules. This is used for highly important messages that need to be sent to the customer as fast as possible.

Field	Type	Description
Create Activities On Activity Date	Bool	<p>If true, then the activities are created at the moment when the messages are sent to the recipients. If false, then the activities are created at the moment when the message are scheduled to be sent to the recipients.</p> <p>For example, if the Type of delay for weekend field is set to Send After and the messages are scheduled to be sent over the weekend, then:</p> <ul style="list-style-type: none"> when set to true, the activities are created on the first day after the weekend (the day when the messages are sent to the recipients). when set to false, then the activities are created in the weekend (the day when the messages are scheduled to be sent to the recipients) and sent to the recipients the first day after the weekend.
Max Delay Days No	Whole Number	Insert the maximum days available for delay of campaign.

Marketing Anti-spam Settings

To avoid spamming the audience or being unnecessarily disruptive to the clients, a special set of rules and conditions have been created in order to help bank employees choose the best course of action (anti-spam rules).

HINT

The anti-spam rules can be set also after saving the campaign.

For all the different types of communication channels available in FintechOS, certain triggers can be defined. These can include the number of days to delay the communication. Click on the **Insert** button to create a new anti-spam rule. The **Add Antispam Settings** page is displayed. Fill in the following fields:

ADD ANTISPAM SETTINGS

ANTISPAM SETTINGS

Communication Channel ↓ ↗

Source Campaign ↓ ↗

Delay Days No

Field	Type	Description
Communication Channel	Lookup	The communication channel for which the anti-spam rules apply. For example: SMS, email.
Source Campaign	Lookup	The campaign according to which the anti-spam rules are set. The existing campaign does not start until the campaign selected in this field ends.
Delay Days No	Whole Number	The number of days that communications will be delayed in order to prevent spamming the audience with the same message in a short period of time.

Click the **Save and reload** button. The **Edit Campaign** page is displayed. To navigate between sections, click on the bullets from the bullet list which renders the section tabs.

Defining the Campaign Audience

Audience is any specific person that exists in the database that agreed to a data processing consent for marketing purposes and which fulfills any conditions that might be set. To define the campaign audience, click the **Audience** tab. The content of the **Audience** section varies based on the **Audience Type** selected. The Audience field filters the available audience with certain rules data in order to deliver a customizable marketing experience. The data source for the audience is the audience type, which can be:

Imported

An imported list behaves similar to a static list with the exception that it is a list from an outside source for example, a marketing partner. When choosing to import an audience list, the following fields are displayed:

The screenshot shows a form titled "ADD CAMPAIGN". Under "Audience Type", "Imported List" is selected. Below this are fields for "File ID Field Name" and "Audience Name Field". An "Import File" button is present, with a file named "Book1.xlsx" selected. Below the button, it says "Select file or Drop file here".

NOTES:
 For **Dynamic** or **Static** List Audience Type, the audience must provide 2 mandatory tokens: **RECEIVER_EMAIL** and **RECEIVER_PHONE** (even when no data will be provided for one of the tokens)! In case you use the audience in an **Internal Campaigns**, depending on distribution type, the audience must also provide the following 2 tokens: **ACTIVITY_BRANCH** and/or **ACTIVITY_OWNER**.
 For **Imported** List Audience Type, the audience must provide 4 mandatory tokens (excel columns):
 - A column that will represent the unique identification of a record (File ID Field Name)
 - A column for the Audience "Name" Field
 - Two columns called exactly **phone** and **email**, even when no data will be provided for all or some of the cells in the column.
 In case the audience is used in **Internal Campaigns**, the Excel must include five additional columns named exactly **branch**, **owner**, **accountName**, **accountPIN** and **accountFiscalRegistrationNo**, even if no data will be provided for some cells of the column.

Field	Description
Audience Type	The type of audience which can be imported, static, or dynamic.
Audience List	Displayed only for
File ID Field Name	<p>After the audience list is generated, the field is no longer displayed at form level. Based on the information from this field, the system identifies which is the unique ID of each record from the import file. The user needs to fill-in the name of the column (from the imported audience list) that contains the unique ID of the record.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #add8e6;"> <p>NOTE If this field is not completed, the following error message is displayed: File ID Field Name' and 'Audience Name Field' are mandatory!</p> </div>
Audience Name Field	<p>After the audience list is generated, the field is no longer displayed at form level. Based on the information from this field, the system identifies which is the name of each record from the import file. The user needs to fill-in the name of the column (from the imported audience list) that contains the name of the record.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #add8e6;"> <p>NOTE If this field is not completed, the following error message is displayed: File ID Field Name' and 'Audience Name Field' are mandatory!</p> </div>
Import File	After the audience list is generated, the field is no longer displayed at form level. Select or drag and drop the Excel format file with the audience list.

For Imported List audience type, four mandatory Excel columns, or tokens, must be provided for the audience:

- A column that represents the unique identification of a record: File ID Field Name.
- A column for the **Audience Name** field.
- And two columns called exactly "**phone**" and "**email**", even if data is provided for only one of these columns.

IMPORTANT!

For internal campaigns, five additional Excel columns, or tokens, must be provided for the audience. These columns must be called exactly "**branch**", "**owner**", "**accountName**", "**accountPIN**" and "**accountFiscalRegistrationNo**", even if no data is provided for one of the columns.

Static

A static audience list is a list that remains the same over time. When choosing to use a static audience list, the following fields are displayed:

ADD CAMPAIGN

Audience Type: Static List (selected) | Audience List: [input field]

Audience: [dropdown arrow]

NOTES:
 For **Dynamic** or **Static** List Audience Type, the audience must provide 2 mandatory tokens: **RECEIVER_EMAIL** and **RECEIVER_PHONE** (even when no data will be provided for one of the tokens!)
 In case you use the audience in an **Internal Campaigns**, depending on distribution type, the audience must also provide the following 2 tokens: **ACTIVITY_BRANCH** and/or **ACTIVITY_OWNER**.
 For **Imported** List Audience Type, the audience must provide 4 mandatory tokens (excel columns):
 - A column that will represent the unique identification of a record (File ID Field Name)
 - A column for the Audience "Name" Field
 - Two columns called exactly **phone** and **email**, even when no data will be provided for all or some of the cells in the column.
 In case the audience is used in **Internal Campaigns**, the Excel must include five additional columns named exactly **branch**, **owner**, **accountName**, **accountPIN** and **accountFiscalRegistrationNo**, even if no data will be provided for some cells of the column.

Field	Description
Audience Type	The type of audience which can be imported, static, or dynamic.
Audience List	This field is read only. If the audience type is Imported , then the name of the audience list and the imported template is displayed.
Audience	The audience type.

For **Dynamic** and **Static** list audience type, two tokens or columns must be provided for the audience: **RECEIVER_EMAIL** and **RECEIVER_PHONE** even if no data is provided for one of the tokens.

IMPORTANT!

For internal campaigns, one of the following additional Excel columns, or tokens, must be provided for the audience: **ACTIVITY_BRANCH** and **ACTIVITY_OWNER**. These tokens are provided depending on the **distribution type**. The mandatory tokens based on the distribution type are:

- If the distribution type is set to **Assign To Account Responsible**, then the mandatory token is **ACTIVITY_OWNER**
- If the distribution type is set to **Assign To Account Branch**, then the mandatory token is **ACTIVITY_BRANCH**

Dynamic

A dynamic audience list is a list that changes over time.

ADD CAMPAIGN

Audience Type: • Dynamic List

Audience List:

Audience:

NOTES:
 For **Dynamic** or **Static** List Audience Type, the audience must provide 2 mandatory tokens: **RECEIVER_EMAIL** and **RECEIVER_PHONE** (even when no data will be provided for one of the tokens)! In case you use the audience in an **Internal Campaigns**, depending on distribution type, the audience must also provide the following 2 tokens: **ACTIVITY_BRANCH** and/or **ACTIVITY_OWNER**.
 For **Imported** List Audience Type, the audience must provide 4 mandatory tokens (excel columns):
 - A column that will represent the unique identification of a record (File ID Field Name)
 - A column for the Audience "Name" Field
 - Two columns called exactly **phone** and **email**, even when no data will be provided for all or some of the cells in the column.
 In case the audience is used in **Internal Campaigns**, the Excel must include five additional columns named exactly **branch**, **owner**, **accountName**, **accountPIN** and **accountFiscalRegistrationNo**, even if no data will be provided for some cells of the column.

Field	Description
Audience Type	The type of audience which can be imported, static, or dynamic.
Audience List	This field is read only. If the audience type is Imported , then the name of the audience list and the imported template is displayed.
Audience	The audience type.

For **Dynamic** and **Static** list audience type, two tokens or columns must be provided for the audience: **RECEIVER_EMAIL** and **RECEIVER_PHONE** even if no data is provided for one of the tokens.

IMPORTANT!

For internal campaigns, one of the following additional Excel columns, or tokens, must be provided for the audience: **ACTIVITY_BRANCH** and **ACTIVITY_OWNER**. These tokens are provided depending on the [distribution type](#). The mandatory tokens based on the distribution type are:

- If the distribution type is set to **Assign To Account Responsible**, then the mandatory token is **ACTIVITY_OWNER**
- If the distribution type is set to **Assign To Account Branch**, then the mandatory token is **ACTIVITY_BRANCH**

Click the **Save and reload** button. The **Edit Campaign** page is displayed. To navigate between sections, click on the bullets from the bullet list which renders the section tabs.

Scheduling the Campaign in Stages

A campaign can have multiple stages and each stage has a stage instance. Each time a stage executes, a stage instance is created. If it's a one time only campaign, it has one stage, one instance.

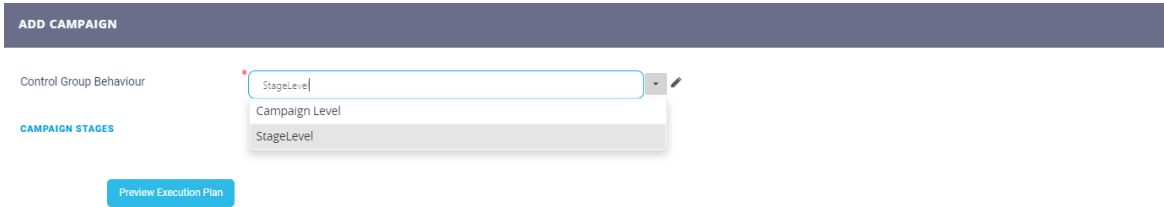
Click the **Schedule** tab and define the starting dates and end dates along with all the campaign stages. New campaign stages can be easily added or modified. Campaign stages have additional options that can be configured and support AB testing.

HINT

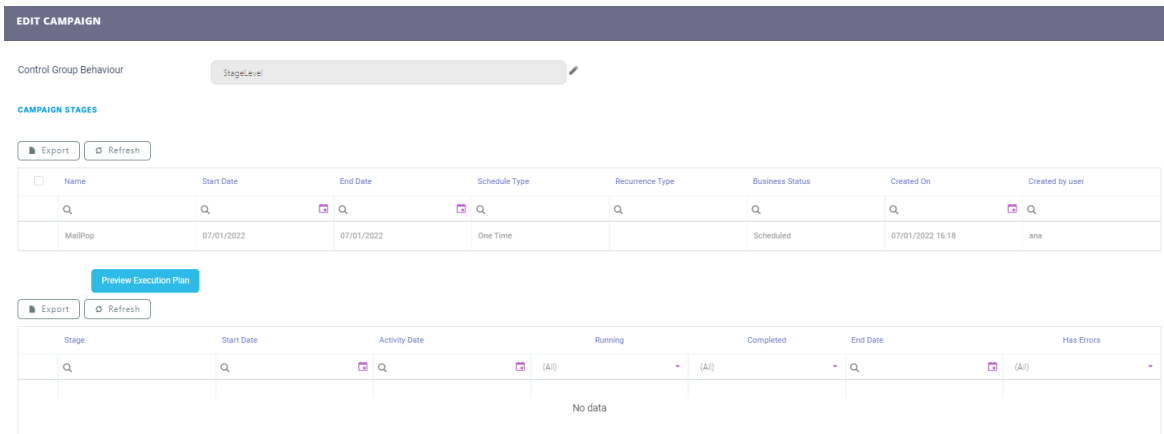
Simple AB tests can also be created. A/B testing (bucket tests or split-run testing) is a randomized experiment with two variants, A and B.

Campaign stages can be scheduled at **Campaign Level** or at **Stage Level** through the **Control Group Behaviour** field.

For stage level campaigns the control group members are kept for the entire campaign and for all stages. For campaign level campaigns the control group members are different for each stage.



Setup



To add a stage, click the **Insert** button from the **Campaign stages** section. The **Add Campaign Stage** page is displayed.

Setting up the campaign stage is done similarly to setting up a campaign. This includes fields for the campaign stage name and the **Content Template**, but also personalized content to be used during the campaign stage.

NOTE
 If at stage level, the communication channel has the **Is Campaign Activity Type** field set to true, then the stage generates internal campaign activities. For more details on communication channel configurations, see the [Create the Channel Configurations](#) page.

Field	Description
Name	Insert a suggestive name for the stage.
Content Template	Select a template for the stage. It can be a different template than the template selected for the whole campaign.
Culture	The campaign culture. For example: RO, GB. The available values are the ones from the selected content template.
Time zone	Select from the list the time applicable.
Activity type	<p>This field is displayed when the content template selected has a communication channel with the Is Campaign Activity Type field set to true. To configure a communication channel , see the Create the Channel Configurations page.</p> <p>The following options are available:</p> <ul style="list-style-type: none"> • Sales • Administrative
Activity subtype	<p>The list of attributes that can be selected is defined by the admin user.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #c0d9ff;"> <p>NOTE This field is displayed if the Activity type field value is set to Administrative.</p> </div>
Convert Activity Option	<p>If true, the Convert button and the Related Journey lookup field are displayed at the Campaign Activity form level. For more information see the Actions Buttons page.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #c0d9ff;"> <p>NOTE This field is displayed if the Activity type field value is Sales.</p> </div>
Status Reasons List	This field is displayed when an activity type is selected. To add a status reasons list, select from the available options or create a new one by following the steps from the Managing Status Reasons page.

Click the **Save and reload** button. The page **Edit Campaign Stage** is displayed.

Schedule

This section tab has configurations which apply to only one stage, the stage the user is creating. The stage configurations are found in the global **Schedule** section tab of the whole campaign.

Moreover, the schedule is fully customizable, you can define a starting and ending date along with a recurrence type. You can also have a different schedule type that allows many different combinations to occur: daily, monthly, every X day, where X can be a number, every specific day combination of the week, and others. Previewing these settings offers a powerful way to set up an in-depth strategy of communication in order to provide your audience with the most effective possible way of receiving content updates.

EDIT CAMPAIGN STAGE

Controlled By Execution Plan

Schedule Type: Campaign Life Time

Refresh Control Group On Recurrence:

Start Date: 07/03/2022 End Date: 14/03/2022

startDateVariant: 07/03/2022 clientTimezoneOffset: 2

Recurrence Type: At every 30 Minutes No Of Minutes: 1

Activity Validity Days: [Empty field]

Preview Execution Plan

Export Refresh

Stage	Start Date	Activity Date	Running	Completed	End Date	Has Errors
Q	Q	Q	(All)	(All)	Q	(All)
No data						

CAMPAIGN STAGE INSTANCES

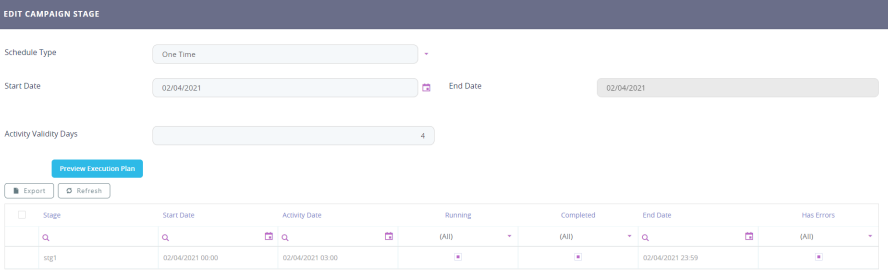
Export Refresh

Stage	Start Date	Activity Date	Running	Completed	Cancel	End Date	Activity Creation Date	Activity Finish Date	Has Errors	Run Log
AnalTestT1	07/03/2022 11:00	07/03/2022 02:00	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	07/03/2022 11:00			<input type="checkbox"/>	Stage completed successfully in 00:00:00
AnalTestT1	07/03/2022 10:30	07/03/2022 02:00	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	07/03/2022 10:30	07/03/2022 10:30		<input type="checkbox"/>	Stage completed successfully in 00:00:12

Field	Description
Controlled By Execution Plan	If true, the fields used to define the stage schedule are hidden and the stage can be started only from an execution plan.

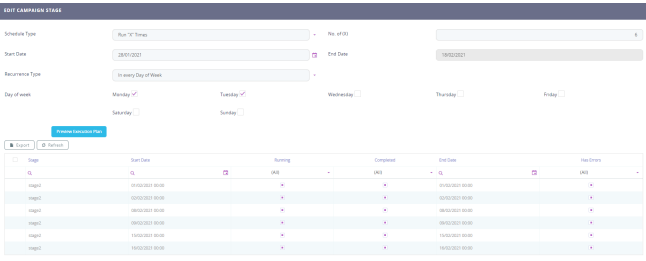
Field	Description
Schedule Type	<p>The type of recurrence that the campaign stage will run on:</p> <ul style="list-style-type: none"> • one time • campaign life time • Fix dates • Run for "X" days • Run for "X" weeks • Run for "X" months • Run for "X" times.
No. of X	<p>Number of recurrences.</p> <div style="background-color: #f4a460; padding: 10px; border-radius: 5px;"> <p>IMPORTANT! Depending on this number you insert, the period of the stage will grow. If it is too big, it may become bigger than the date period for the whole campaign, in which case it will not allow you to save the stage.</p> </div>
Start Date	Starting date for the campaign stage.
End Date	Ending date for the campaign stage.
Recurrence Type	<p>Type of recurrence that will be executed by the campaign stages. Depending on the selection of these types, certain fields appear. Select from the list:</p> <ul style="list-style-type: none"> • at every "X" minutes • at every "X" days • in every day of week • week of month • at day of month • at season concurrence.

AUTOMATION BLOCKS USER GUIDE

Field	Description
Refresh Control Group On Recurrence	<p>This field is not displayed if the recurrence interval type is set to One Time.</p> <p>If set to <code>true</code>, then the control group is refreshed before every recurrence of the stage.</p> <p>If set to <code>false</code>, then the control group is the same for all stage recurrences.</p>
Activity Validity Days	<p>By default, activities generated in a stage are active for one day before they expire. If this time frame is not sufficient, you can manually enter the number of days you wish to keep activities valid.</p>  <p>The screenshot shows the 'EDIT CAMPAIGN STAGE' configuration page. The 'Schedule Type' is set to 'One Time'. The 'Start Date' is 02/04/2021 and the 'End Date' is 02/04/2021. The 'Activity Validity Days' is set to 4. Below the form is a table with columns: Stage, Start Date, Activity Date, Running, Completed, End Date, and Has Errors. The table contains one row for stage 'mg1' with start date 02/04/2021 00:00, activity date 02/04/2021 03:00, and end date 02/04/2021 23:59.</p>
Client Timezone Offset	The client's time zone. This field is read only.

For each type of recurrence a new field or fields open.

Type of recurrence	Fields	Description
At every X Days	No of X	How many days.
	Daily No of Days	The gap between those X, e.g. No of X 3 and Daily No of Days 5, renders three days when the campaign is sent with 5 days in between each of those three days.
At every x Minutes	No of X	How many minutes.
	No of Minutes	The gap between those minutes.

Type of recurrence	Fields	Description
In every day of week	No of X	The number of days.
	Checkboxes with Weekdays	The week days: Monday to Sunday. 
Week of Month	No of X	The number of weeks.
	Week No (1-5)	In which week the month the campaign will be executed.
At day of month	No of X	The number of days.
	Day No.	the actual calendar number of the day, from 1 to 31. e.g. 4 will render every day with the calendar structure 04/m/y.
At season occurrence	No of X	number of seasons.
	Season	Select one of the seasons: <ul style="list-style-type: none"> • Spring • Summer • Autumn • Winter

Click the **Save and reload** button. By clicking on the **Preview Execution plan**, the grid below the button populates with stage instances simulations in order to have a perspective on how the stage instances run.

If the stage is set to run one time, only one stage instance is displayed while for stages set to run with a certain recurrence, the stage instances are created based on how many times the stage runs.

IMPORTANT!
 This is the plan for one stage only. To see the plan of all stages return to the **Schedule** section tab of the whole campaign.

AUTOMATION BLOCKS USER GUIDE

Preview Execution Plan

Export Refresh

Stage	Start Date	Activity Date	Running	Completed	End Date	Has Errors
Stage_2	03/05/2021 00:00	03/05/2021 02:00	(All)	(All)	03/05/2021 00:00	(All)
Stage_2	07/06/2021 00:00	07/06/2021 02:00			07/06/2021 00:00	
Stage_2	05/07/2021 00:00	05/07/2021 02:00			05/07/2021 00:00	

Campaign Stage Instances (Actual Run)

At stage level, this section populates once the campaign is in an **Approved** business status and when a stage instance is created for every stage run, showing the actual history of the stage instances that were run, not only a simulated preview. If the stage has a recurrence set, then a stage instance is created for every time that recurrence runs.

1 Setup 2 Schedule 3 Distribution 4 Define A/B Variations

EDIT CAMPAIGN STAGE

Controlled By Execution Plan

Schedule Type: One Time

Start Date: 24/03/2022 End Date: 24/03/2022

startDateInvariant: 24/03/2022 clientTimezoneOffset: 1

Activity Validity Days

Preview Execution Plan

Export Refresh

Stage	Start Date	Activity Date	Running	Completed	End Date	Has Errors
Q	Q	Q	(All)	(All)	Q	(All)
No data						

CAMPAIGN STAGE INSTANCES

Export Refresh

Stage	Start Date	Activity Date	Running	Completed	Cancel	End Date	Activity Creation Date	Activity Finish Date	Has Errors	Run Log
sp1	24/03/2022 16:05	24/03/2022 02:00	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	24/03/2022 16:05	24/03/2022 16:05		<input type="checkbox"/>	Stage completed successfully in 00:00:03

To access additional information, double-click on a record. Three section tabs open.

Stage Instance

On this section tab, the following information can be found:

AUTOMATION BLOCKS USER GUIDE

1 Stage Instance
2 Campaign Activities
3 Execution Errors

CAMPAIGN STAGE INSTANCE

Campaign	AnaTest1	Stage	AnaTest1
Start Date	07/03/2022 10:30	End Date	07/03/2022 10:30
Multi Stage Plan Instance	ANA test 7/03 - 2022-09-07 08:29		
IsPreview	<input type="checkbox"/>	Delivery status	Allowed
Running	<input type="checkbox"/>	Cancelled	<input type="checkbox"/>
Completed	<input checked="" type="checkbox"/>	Sent	<input type="checkbox"/>

Suspend delivery
Cancel processing

STAGE INSTANCE RUNNING COUNTERS

Export
Refresh

Channel	Content Template Item	Sent	Total Activities	Total Actions	Total Campaign Activities	Queue Messages Expired	Queue Messages Sent	Queue Messages with Errors	Not Sent
SMS	MPMSMS	2	0	2	0	0	0	0	0
email	content MPM	2	2	2	0	0	0	1	0

Field	Description
Campaign	The name of the campaign.
Stage	The name of the stage.
Start Date	The date the stage began running.
End Date	The date the stage stopped running.
Multi Stage Plan Instance	<p>The name of the multi-stage execution plan instance composed from the multi-stage execution plan name and the time the instance began to run.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #007bff; margin: 10px 0;"> <p>NOTE The Multi Stage Plan Instance field indicates if the stages instance is created at campaign level or if it's part of an execution plan.</p> </div>
IsPreview	If true, it indicates that the stage instance has been generated based on an execution plan preview. For more details on this topic, see the Previewing the Execution Plan page.
Delivery status	<p>The delivery status. The following options are available:</p> <ul style="list-style-type: none"> Suspended: this status is displayed when the Suspend delivery button is used. Allowed: this status is displayed when any other button than Suspend delivery.
Running	If true , the multi-stage stage execution plan is running.
Cancelled	If true , the multi-stage execution plan is cancelled.
Completed	If true , all the activities at stage instance level are created.

Field	Description
Sent	If true , all the messages generated by the stage instance are sent to the recipients.

Here, campaign managers have the possibility to pause, resume, or cancel the delivery of the messages generated by a stage instance by using the below options:

- **Suspend delivery:** Use this button to stop the delivery of all the remaining messages that have not been sent to the recipient on that stage instance. In this case, messages are still created but are no longer sent to the recipients.

This button is displayed if the **Sent** field is set to `false` and the **Delivery Status** field value is **Allowed**.

- **Resume delivery:** Use this button to resume the delivery of the messages that haven't been sent to the recipient.

This button is displayed if the **Delivery Status** field value is **Suspended**.

- **Cancel processing:** Use this button to cancel the creation and delivery of the messages that have not been created or sent to the recipient. This button is displayed if the **Sent** field is set to `false`.

The **Stage Instance Running Counters** section indicates the total number of communications sent to the recipients, the channels used, if the messages were delivered, the queued messages sent, and so on.

Campaign Activities

This section tab displays the activities and action items created and at stage instance level.

AUTOMATION BLOCKS USER GUIDE

1 Stage Instance
2 Campaign Activities
3 Execution Errors

CAMPAIGN ITEMS

Refresh

Stage	Identity	Date	End Date	Email	Phone Number	Channel	Status	Message Status	Message Template	Name	Not Sent	Not Sent Reason	Channel/Provider/Params	List Member	CC Email Address	BCC Email Address	Account Name	PKN
msg.MBInew	8238134556	21/03/2022 02:00	22/03/2022 02:00	dm_l@igbank.by	0887777888	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			ДИМИТЪР ПЕТРОВ ИВАНОВСКИ			ДИМИТЪР ПЕТРОВ ИВАНОВСКИ	8238134556
msg.MBInew	7642426018	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	084423265	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			ДОБРИНКА АНГЕЛОВА СТАСОВА			ДОБРИНКА АНГЕЛОВА СТАСОВА	7642426018
msg.MBInew	8103104455	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	084423265	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			КРИСАНТЕМА МИЛЕНКОВА			КРИСАНТЕМА МИЛЕНКОВА	8103104455
msg.MBInew	905798134880	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0888420810	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			ИОН ИВ ДИМИТЪР ВЪЛЧЕВ ВЪЛЧЕВ			ИОН ИВ ДИМИТЪР ВЪЛЧЕВ ВЪЛЧЕВ	
msg.MBInew	254893209	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0876367217	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			ТЕОДО КОБАЛ ПЛОС ЕООД			ТЕОДО КОБАЛ ПЛОС ЕООД	254893209
msg.MBInew	113035443	21/03/2022 02:00	22/03/2022 02:00	netem@gmail.by	088734248	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			ВАНТО ТРЕДЪ АПТО АД			ВАНТО ТРЕДЪ АПТО АД	113035443
msg.MBInew	203374133	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0876367217	Mailbox_Bulk message	InProgress	Sent	MBInew_3	MBInew_3	<input type="checkbox"/>			АПС БЕТА БЪЛГАРИЯ ЕООД			АПС БЕТА БЪЛГАРИЯ ЕООД	203374133

ACTIVITIES

Refresh

Stage	Identity	Date	End Date	Email	Phone	Channel	Status	Message Status	Message Template	Assign User	Not Sent	Not Sent Reason	Channel/Provider/Params	List Member	Name	Account Name	PKN	Facial Registration No	Comments
msg.MBInew	204893209	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0876367217	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			ТЕОДО КОБАЛ ПЛОС ЕООД	MBInew_3	ТЕОДО КОБАЛ ПЛОС ЕООД	254893209	254893209	254893209
msg.MBInew	8238134556	21/03/2022 02:00	22/03/2022 02:00	dm_l@igbank.by	0887777888	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			ДИМИТЪР ПЕТРОВ ИВАНОВСКИ	MBInew_3	ДИМИТЪР ПЕТРОВ ИВАНОВСКИ	8238134556	N/A	N/A
msg.MBInew	7642426018	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	084423265	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			ДОБРИНКА АНГЕЛОВА СТАСОВА	MBInew_3	ДОБРИНКА АНГЕЛОВА СТАСОВА	7642426018	N/A	N/A
msg.MBInew	905798134880	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0888420810	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			ИОН ИВ ДИМИТЪР ВЪЛЧЕВ ВЪЛЧЕВ	MBInew_3				
msg.MBInew	8103104455	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	084423265	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			КРИСАНТЕМА МИЛЕНКОВА	MBInew_3	КРИСАНТЕМА МИЛЕНКОВА	8103104455	N/A	N/A
msg.MBInew	203374133	21/03/2022 02:00	22/03/2022 02:00	info@igbank.by	0876367217	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			АПС БЕТА БЪЛГАРИЯ ЕООД	MBInew_3	АПС БЕТА БЪЛГАРИЯ ЕООД	203374133	203374133	203374133
msg.MBInew	113035443	21/03/2022 02:00	22/03/2022 02:00	netem@gmail.by	088734248	Mailbox_Bulk message	Planned	Sent			<input type="checkbox"/>			ВАНТО ТРЕДЪ АПТО АД	MBInew_3	ВАНТО ТРЕДЪ АПТО АД	113035443	113035443	113035443

CONTROL GROUP

Export Refresh

Stage	Activity Date	List Member	Entry Type	Customer
No data				

Execution Errors

This section tab shows if any errors occurred while running the execution plan.

1 Stage Instance
2 Campaign Activities
3 Execution Errors

Has Errors

Error Message

Distribution

IMPORTANT!


The campaign stage distribution step is displayed only for internal campaigns. A campaign that generates internal campaign activities cannot be approved without selecting the distribution type.

This tab allows campaign managers to set up a distribution logic when creating an internal campaign. Thus, having the flexibility to assign the campaigns activities either to certain responsible persons or to certain branches.

AUTOMATION BLOCKS USER GUIDE

EDIT CAMPAIGN STAGE

Distribution Type

Custom Assignment Logic 

Assignment Logic

Distribute Uniform to Branch Users 

Field	Description
Distribution Type	<p>Select the target responsible for the campaign activities. The following options are available:</p> <ul style="list-style-type: none"> • Assign To Account Responsible: in this case, the activity is distributed to the account responsible. For static and dynamic audience lists, each activity is assigned to the account owner. For imported audience lists, each activity is assigned to the account responsible from the imported audience list. • Assign To Responsible Branch: in this case, the activity is distributed to the responsible branch. For static and dynamic audience lists, each activity is assigned to the responsible branch responsible. For imported audience lists, each activity is assigned to the account responsible branch from the imported audience list. • Custom Assignment Logic: in this case, the activity is distributed to a user from the responsible branch. This is a custom stored procedure. For static and dynamic audience lists, each activity is assigned to the account responsible, based on a logic that is defined at the time of implementation. For imported audience lists, each activity is assigned to the account responsible from the imported audience list, based on a logic that is defined at the time of implementation. <p>This option in the distribution logic gives the possibility of assigning activities to the person responsible considering their competencies and workload based on the following scenarios:</p> <ul style="list-style-type: none"> • Assign activities to users from the activity responsible branch ensuring a uniform random distribution per campaign. Activities are allocated equally to all team members, taking into account the activities from the

Field	Description
	<p>current campaign.</p> <ul style="list-style-type: none"> • Assign activities to users from the activity responsible branch that have specific role, function, or competence (for example relationship manager, teller, account manager) ensuring a uniform random distribution per campaign. Activities must be allocated equally to all team members, taking into account the activities from the current campaign. • Assign activities to users from the activity responsible branch that have specific role, function, or competence (for example relationship manager, teller, account manager) ensuring a uniform random distribution taking into account the number of all open (activities with the In progress or Assigned status) campaign activities already allocated. In this case, the activities are allocated equally to all team members, taking into account the open activities of all ongoing campaigns.

Field	Description
Assignment Logic	<p>This field is displayed only for Custom Assignment Logic distribution type. The assignment logic runs for each internal campaign activity when the campaign manager or segment manager selects the Custom Assignment Logic option and uses one of the available custom stored procedures.</p> <p>The following parameters are taken into account when defining the stored procedures:</p> <ul style="list-style-type: none"> • activity responsible branch, • users from responsible branch, • users role, function, or competence, • number of activities that are already allocated per current campaign, • number of activities already allocated with In Progress or Assigned status, per all in progress campaigns.

Click the **Save and Reload** button.

The following warning message is displayed if the **Distribution Type** field is not set in the following instances: Before approving the campaign, select the distribution type for the stage: "stage name". This message appears in the below instances:

- If the field is not set for at least one stage that generates internal campaign activities.
- For a campaign with two stages where one generates internal activities and the other external ones, if both stages do not have the field set, the same message is displayed, showing the name of the internal campaign stage.

The message is not displayed in the below situations:

- For campaigns that generate only external campaign activities.
- For campaigns with two stages, a stage with internal activities and one with external ones, and where the field is set for the stage that generates internal activities.

HINT

Stored procedures templates can be modified or new ones can be added at product implementation, depending on the customer's business needs.

Define A/B variations of a stage

Depending on the template created and selected for the campaign, certain channels are rendered here:

- email
- SMS.

This tab contains the following fields:

Field	Description
Allow recurrence Duplicates	If true, it allows any recurring messages to be duplicates of previous messages.
Channel Variations list	The proportion for sending differentiated communications.
Content and Channel Variations list	The proportion for sending differentiated communications of the same type.

EDIT CAMPAIGN STAGE

Allow Recurrence Duplicates

CHANNEL VARIATIONS

<input type="checkbox"/> Channel	Send Channel (%)	Start Time (hh)	Start Time (mm)	End Time (hh)	End Time (mm)
EMAIL MFA	100.00	00	00	23	59
CONTROL GROUP	0.00	00	00	00	00

CONTENT & CHANNEL VARIATIONS

<input type="checkbox"/> Channel	Template	Send Template (%)
No data		

IMPORTANT!

These variations apply to only one stage.

Edit the:

Field	Description
Send channel %	There is the percentage used for the usage of the channel.
Start time (hh)	The starting hour.

Field	Description
Start time (mm)	The starting minutes.
End time (hh)	The ending hour.
End time (mm)	The ending minute.

Example

A company with a customer database of 2,000 people decides to create an email campaign with a discount code in order to generate sales through its website. It creates two versions of the email with different call to action (the part of the copy which encourages customers to do something — in the case of a sales campaign, make a purchase) and identifying promotional code. To 1,000 people it sends the email with the call to action stating, "Offer ends this Saturday! Use code A1", and to another 1,000 people it sends the email with the call to action stating, "Offer ends soon! Use code B1".

All other elements of the emails copy and layout are identical. The company then monitors which campaign has the higher success rate by analyzing the use of the promotional codes.

The email using the code A1 has a 5% response rate (50 of the 1,000 people emailed used the code to buy a product), and the email using the code B1 has a 3% response rate (30 of the recipients used the code to buy a product).

The company therefore determines that in this instance, the first Call to Action is more effective and will use it in future sales.

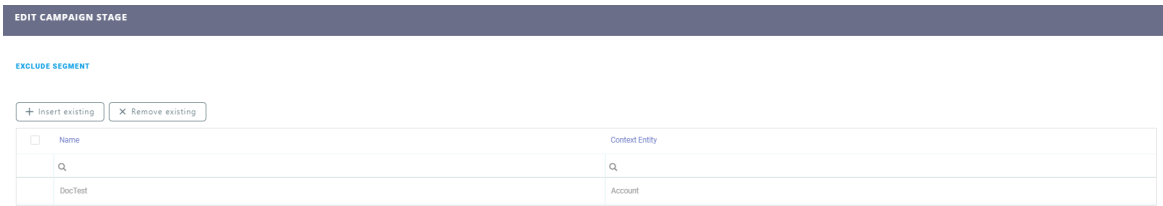
A more nuanced approach would involve applying statistical testing to determine if the differences in response rates between A1 and B1 were statistically significant (that is, highly likely that the differences are real, repeatable, and not due to random chance).

There are many A/B testing case studies which show that the practice of testing is increasingly becoming popular with small and medium-sized businesses as well.

You can define send channel percentages in order to distinguish between different channels. The view to be sent can be previewed along with the starting and end times. Further differences might be set in order to differentiate between two versions of the same channel communication. For example, you can have two versions of the same email and split those to determine the results.

Click the **Save and reload** button.

Refining Audience of a stage



Configuring the selected audience can be further changed from the **Refining Audience** tab, by excluding specific segments from the initial data set available. These are a set of conditions that will further filter the selected audience (examples include: customers under 20, customers without children, etc.).

From the audience selected it is possible to eliminate certain people depending. Click the "Insert existing" and select from the pop-list the audience which will not receive the campaign messages.

HINT
Create the list with the audience to be excluded before arriving to this step.

Activities of a stage

This tab contains the actions, activities, and campaign activities generated at stage level.

Actions

ACTIONS

Refresh

Stage	Identity	Date	End Date	Email	Phone Number	Channel	Status	Message Status	Message Template	Name	Not Sent	Not Sent Reason	Channel/Provider/Params	List Me
test0702	SQL3645184	10/02/2022 02:00	11/02/2022 02:00	hard drive_66@Daniel - Feeney.com		TCA	InProgress	New	content11item	content11item	<input type="checkbox"/>			Chandi
test0702	IB2130955	10/02/2022 02:00	11/02/2022 02:00	hard drive_66@Schroeder, Miller and Barrows.com	1-276-995-0099 x07837	TCA	InProgress	New	content11item	content11item	<input type="checkbox"/>			Anna N
test0702	SSL2648749	10/02/2022 02:00	11/02/2022 02:00	hard drive_66@Halvorson and Sons.com		TCA	InProgress	New	content11item	content11item	<input type="checkbox"/>			Macey
test0702	SMT19306727	10/02/2022 02:00	11/02/2022 02:00	hard drive_66@Bosco- Cole.com	602-736-7722 x7794	TCA	InProgress	New	content11item	content11item	<input type="checkbox"/>			Maybe

Actions can be used (**Actions** tab) to automatically send email or SMS messages, based on user configuration, and place them in a system queue which will then deliver them to the intended audience. The difference between actions and activities is that activities are not sent to the audience.

Activities

ACTIVITIES

Export Refresh

Stage	Identity	Date	End Date	Email	Phone	Channel	Status	Message Status	Message template	Assigned User
Stage 1	pixel_44@Feesat.com	10/09/2021 14:45	11/09/2021 14:45	pixel_44@Feesat.com	716-295-5209	TCA	Planned			
Stage 1	system_42@Comman.com	10/09/2021 14:45	11/09/2021 14:45	system_42@Comman.com	450-334-6300	TCA	Planned			
Stage 1	sensor_40@Comman.com	10/09/2021 14:45	11/09/2021 14:45	sensor_40@Comman.com	1-403-334-6300	TCA	Planned			

Activities can be viewed at campaign or stage level. In this tab, all the campaign activities generated at stage level are displayed.

Campaign Activities

CAMPAIGN ACTIVITIES

Export Refresh

Activity Type	Activity Subtype	Stage	Account PIN	Account Name	Activity status
Sales		stage1newtestintern		Maybell	Cancelled
Sales		stage1newtestintern		Chandler	Cancelled
Sales		stage1newtestintern		Anna	Cancelled
Sales		stage1newtestintern		Macey	Cancelled

This section contains the internal campaign activities generated within the stage level. For more details see the [Internal Campaign Activities](#) page.

A/B Control Group of a stage

EDIT CAMPAIGN STAGE

ACTIONS

Export Refresh

Stage	Activity Date	List Member	Entity Type	Customer
Stage 1	10/09/2021 03:00	Alford	Account	Alford
Stage 1	10/09/2021 03:00	Caterina Lopez	Account	Caterina Lopez
Stage 1	10/09/2021 03:00	Kole Clark	Account	Kole Clark

This section tab displays the messages or actions kept at control group level and not sent to the targeted audience.

Execution Log of a stage

The **Execution Log** tab displays a list of all the actions undertaken in this category along with the associated stage and ending dates. This allows you to easily track and see what happens under the system.

EDIT CAMPAIGN STAGE									
<input type="button" value="Export"/> <input type="button" value="Refresh"/>									
Stage	Start Date	Activity Date	Running	Completed	End Date	Activity Creation Date	Activity Finish Date	Has Errors	Run Log
test	03/02/2022 14:12	03/02/2022 02:00	<input type="checkbox"/>	<input checked="" type="checkbox"/>	03/02/2022 14:12	03/02/2022 14:12		<input type="checkbox"/>	Stage completed successfu...

Previewing the Execution Plan

When the user is done adding all the stages, he/ she returns to the **Schedule** section tab of the whole campaign and clicks on **Preview execution plan** to see the plan for all stages.

EDIT CAMPAIGN							
CAMPAIGN STAGES							
<input type="button" value="+ Insert"/> <input type="button" value="X Delete"/> <input type="button" value="Export"/> <input type="button" value="Refresh"/>							
Name	Start Date	End Date	Schedule Type	Recurrence Type	Business Status	Created On	Created by user
stg1	02/04/2021	02/04/2021	One Time		Scheduled	02/04/2021 10:41	host
Stage_2	11/04/2021	11/07/2021	Run "X" Times	Every "X" Day of Month	Scheduled	02/04/2021 14:39	host

<input type="button" value="Export"/> <input type="button" value="Refresh"/>									
Stage	Start Date	Activity Date	Running	Completed	End Date	Has Errors			
stg1	02/04/2021 00:00	02/04/2021 03:00	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	02/04/2021 23:59	<input checked="" type="checkbox"/>			
Stage_2	03/05/2021 00:00	03/05/2021 02:00	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/05/2021 00:00	<input checked="" type="checkbox"/>			
Stage_2	07/06/2021 00:00	07/06/2021 02:00	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	07/06/2021 00:00	<input checked="" type="checkbox"/>			
Stage_2	05/07/2021 00:00	05/07/2021 02:00	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	05/07/2021 00:00	<input checked="" type="checkbox"/>			

Preview campaign stage instances

Once you have defined and saved a campaign stage, you can generate an Excel file that lists all the data for an instance of that stage. The preview is generated asynchronously by a job scheduler and you have the option to receive an email notification when the Excel file is ready. To create a stage instance preview:

1. In Innovation Studio, open a campaign and select the **Schedule** tab.
2. If the execution plan is not displayed, click the **Preview Execution Plan** button to generate it.
3. Double click a stage instance from the execution plan to open it. Four section tabs open. The fields are read-only.

AUTOMATION BLOCKS USER GUIDE

1 Stage Instance **2** Campaign Activities **3** Activities Preview List **4** Execution Errors

CAMPAIGN STAGE INSTANCE

Campaign: ✓ Stage: ✓

Start Date: End Date:

IsPreview: Completed:

Running:

STAGE INSTANCE RUNNING COUNTERS

<input type="checkbox"/>	Channel	Content Template Item	Sent	Not Sent

No data

4. In the stage instance form, select the **Activities Preview List** tab.
5. Click **Insert** to create a new stage instance preview.
6. In the Add Generated Preview window, enter a **Begin Date** for when you wish to schedule the processing for the preview and select the **Send Mail** checkbox if you wish to send an email notification when the preview is ready.

ADD GENERATED PREVIEW

GENERATED PREVIEW

Begin Date: 📅

Send mail:

7. Click **Save and Reload** at the top right corner of the page.
8. In the Edit Generated Preview page, you can modify the **Begin Date** and **Send Mail** options and customize the **Email Address** where you wish to receive the notification

once the preview is ready.

EDIT GENERATED PREVIEW

GENERATED PREVIEW

Name	activity_CampaignDinamicET1_stg2_05-19-2021_14:16:19
Status	Scheduled ✎
Preview Type	Activity ✎
Begin Date	19/05/2021 14:25 📅
End Date	
Found Records	
Send mail	<input checked="" type="checkbox"/>
Email Address	john.doe@fintechos.com
File	
Campaign	CampaignDinamicET1 ↓ ✎
Stage	stg2 ↓ ✎
Has Errors	<input type="checkbox"/>
Error Message	

Cancel

9. Click **Save and Close** at the top right corner of the page.

To cancel a preview, click on the **Cancel** button found on the Generated preview and confirm.

The Activities Preview List tab lists all the stage instance previews tasks that have been finished, are in progress, have failed, or are scheduled to run in the future.

AUTOMATION BLOCKS USER GUIDE

1 Stage Instance 2 Campaign Activities 3 Activities Preview List 4 Execution Errors

GENERATED PREVIEWS

[+ Insert](#) [X Delete](#) [Export](#) [Refresh](#)

<input type="checkbox"/>	Name	Status	Begin Date	End Date	Preview Type	Found Records	Created On	Has Errors
<input type="checkbox"/>	activity_Campaign...	Finished	30/04/2021 11:16	30/04/2021 11:16	Activity	19	30/04/2021 11:16	<input checked="" type="checkbox"/>
<input type="checkbox"/>	activity_Campaign...	Finished	19/05/2021 14:14	19/05/2021 14:14	Activity	19	19/05/2021 14:13	<input type="checkbox"/>
<input type="checkbox"/>	activity_Campaign...	Scheduled	19/05/2021 14:25		Activity		19/05/2021 14:17	<input checked="" type="checkbox"/>

Once a preview generation is finished, if you select it from the list, you will be able to download the Excel file containing stage instance preview data.

EDIT GENERATED PREVIEW

GENERATED PREVIEW

Name	activity_CampaignDinamicET1_stg2_05-19-2021_14:16:19
Status	Finished <input type="text"/>
Preview Type	Activity <input type="text"/>
Begin Date	19/05/2021 14:25 <input type="text"/>
End Date	19/05/2021 14:25 <input type="text"/>
Found Records	<input type="text"/> 19
Send mail	<input checked="" type="checkbox"/>
Email Address	john.doe@fintechos.com <input type="text"/>
File	FTOS_MKT_StageInstance.xlsx
Campaign	CampaignDinamicET1 <input type="text"/>
Stage	stg2 <input type="text"/>
Has Errors	<input type="checkbox"/>
Error Message	<input type="text"/>

Controlling Campaign Activities

This section allows you to control the associated campaign activities and actions that can be sent. These behave in a similar fashion to the activities and actions configured for the campaign stages only these should occur once the campaign has ended. This feature is used to record the person responsible for sending specific messages and keeping a log of the number of messages sent. In addition, personal information such as name, email, and phone number are also stored along with an associated user to allow an easy way of contacting that person and to promote ownership.

EDIT CAMPAIGN

Control Message Count Random Control Message

CAMPAIGN TEAM MEMBERS

Name	First Name	Last Name	Attached User	Team Member Function	Phone	Email
a	a	a	a	a	a	a
No data						

Campaign Team Members

Team members can be added to the campaign team member list by allowing them to be entered in the **Control** tab. The Control section allows you to select the number of messages that will be sent and varying the control message by selecting the random message option. Each message number is associated to a campaign user.

Field	Description
Control Message Count	The maximum number of messages that are sent to the control group members, if the team members are defined at the campaign level.
Random Control Message	If true, and the control message count is smaller than the number of the messages that the system generates in case the Control Message field is null, then the system randomly selects the messages that are sent to the control group members.

HINT

In the **Campaign Team Members** section, it is possible to select the member who can control the campaign.

Viewing Activities

The **Activities** tab has four sections: **Actions**, **Activities**, **Internal Campaign Activities**, and **Control Group** for an overview of the campaign. In this step, the generated activities at campaign level are displayed.

EDIT CAMPAIGN

ACTIONS

Export Refresh

Stage	Identity	Date	End Date	Email	Phone Number	Channel	Status	Message Status	Message Template	Name	Not Sent	Not Sent Reason	List Member	CC Email Addresses	BCC Email
stp2	59L248749	19/01/2022 02:00	20/01/2022 02:00	hard.drive_11@halverson and Sons.com		SMS	InProgress	New	TEST_A	TESTACTION	<input type="checkbox"/>		Marcy Miles Tabitha Nicolas		
stp2	50L3045184	19/01/2022 02:00	20/01/2022 02:00	hard.drive_74@Denial - Feeney.com		SMS	InProgress	New	TEST_A	TESTACTION	<input type="checkbox"/>		Chandler Kole Kalli Jaskolski		
stp2	5MTP9395727	19/01/2022 02:00	20/01/2022 02:00	hard.drive_34@Bosco- Cole.com	632-736-7722 x7794	SMS	InProgress	New	TEST_A	TESTACTION	<input type="checkbox"/>		Maybell Jonathan Suzanne Hartmann		
stp2	1B2139955	19/01/2022 02:00	20/01/2022 02:00	hard.drive_34@Schroeder, Miller and Barrows.com	1-278-595-0099 x07837	SMS	InProgress	New	TEST_A	TESTACTION	<input type="checkbox"/>		Anna Mayra Steve Hayes		

ACTIVITIES

Export Refresh

Stage	Identity	Date	End Date	Email	Phone	Channel	Status	Message Status	Message Template	Assigned User	Not Sent	Not Sent Reason	List Member	Name	Account Name	PIIN	Facial Registration No	Commercial Registration
No data																		

INTERNAL CAMPAIGN ACTIVITIES

+ Insert X Delete Export Refresh

Activity Type	Activity Subtype	Stage	Account PIN	Account Name	Activity Status
No data					

CONTROL GROUP

Export Refresh

Stage	Activity Date	List Member	Entity Type	Customer
No data				

Actions

This section displays the actions generated through a campaign. The following data is displayed:

Field	Description
Stage	The campaign stage.
Identity	The unique campaign identifier.
Date	The campaign actions start date.
End Date	The campaign actions end date.
Email	The client's email address.
Phone Number	The client's phone number.
Channel	The communication channel.
Status	The campaign status: Scheduled, In progress, Notified, Cancelled, Error.

AUTOMATION BLOCKS USER GUIDE

Field	Description
Message Status	The message status.
Message Template	The message template.
Name	The campaign number.
Not Sent	If selected, the campaign messages are not to be sent for this account.
Not Sent Reason	The reason why messages are not to be sent for this account.
List Member	The audience list member.
CC Email Addresses	Alternative email address to be added in the CC field.
BCC Email Addresses	Alternative email address to be added in the BCC field.
Account Name	The name of the account.
PIN	The account's PIN number.
Fiscal Registration No	The fiscal registration number of a corporate account.
Commercial Registration	The commercial registration number.

Activities

This section displays the activities generated through a campaign. The following data is displayed:

Field	Description
Stage	The campaign stage.
Identity	The unique campaign identifier.
Date	The campaign activities start date.
End Date	The campaign activities end date.
Email	The client's email address.
Phone Number	The client's phone number.
Channel	The communication channel.
Status	The campaign status: Scheduled, In progress, Notified, Cancelled, Error.
Message Status	The message status.
Message Template	The message template.
Assigned User	The user the campaign activity is assigned to.
Not Sent	If selected, the campaign messages are not to be sent for this account.
Not Sent Reason	The reason why messages are not to be sent for this account.
List Member	The audience list member.
Name	The campaign number.
Account Name	The account name.

Field	Description
PIN	The account's PIN number.
Fiscal Registration No	The fiscal registration number of a corporate account.

Internal Campaign Activities

This section displays sales or administrative internal campaign activities.

NOTE

The **Internal Campaign Activities** section is populated only if a campaign generates internal campaigns activities. For additional details, see the [Internal Campaign Activities](#) page.

Field	Description
Activity Type	The activity type: sales or administrative.
Activity Subtype	The activity subtype.
Activity status	The activity status.
Account Name	The name of the account.
Account PIN	The account PIN.
Stage	The campaign stage.

Control Group

This section displays the control group team members. The following data is displayed:

Field	Description
Stage	The campaign stage.
Activity Date	The date when the activity is created.
List Member	The audience list member.
Entity Type	The entity type.
Customer	The customer name.

Saving the Campaign

Before saving the campaign, go to the **Generated Previews** tab to preview it. The following data is displayed:

Name	Status	Begin Date	End Date	Preview Type	Found Records	Created On
activity_Campaign 1_Stage 1_09...	Finished	10/09/2021 11:36	10/09/2021 14:36	Activity	540	10/09/2021 14:36
activity_Campaign 1_Stage 1_09...	Finished	10/09/2021 11:37	10/09/2021 14:37	Activity	540	10/09/2021 14:37

Field	Description
Name	The campaign name.
Status	The campaign status.
Begin Date	The date the campaign started.
End Date	The date the campaign ended.
Preview Type	The preview type. Possible values are: segment, audience, activity, campaign plan.
Found Records	The number of records found.
Created on	The system date when the campaign record is inserted.

To save the campaign, click the **Save and Close** button at the top right corner.

Internal Campaigns

Campaigns are made through typical communication channels namely email, text messaging, online chat messaging, phone calls, mail, or even through the banking mobile application that the customer is using. Once these campaigns are started, they generate certain campaign activities that need to be handled internally.

The **Internal Campaigns** module comes as an add-on to the Omnichannel Campaigns automation processor and it's meant as an extension to the current customer interaction channels. Along with email, text messaging, online messaging, and other means of communication, there are times when human interaction is needed, especially in certain sales actions. Besides sales activities, administrative actions can be assigned and managed also through this module. In these situations, a bank employee reaches out to a client with administrative purposes, for example, data or document collection.

These types of internal campaign activities are distributed to internal roles: relationship manager, account manager, contract center agent, teller, and so on.

HINT

Campaigns generate internal or external activities based on the communication channel set at the content used at stage level. When defining a communication channel, the **Default Action Type** field must be set. If at **Action Type** level, the **Is Campaign Activity Type** field is true, then the campaign generates internal activities. For more information on adding communication channels, see the [Managing Personalized Content](#) page.

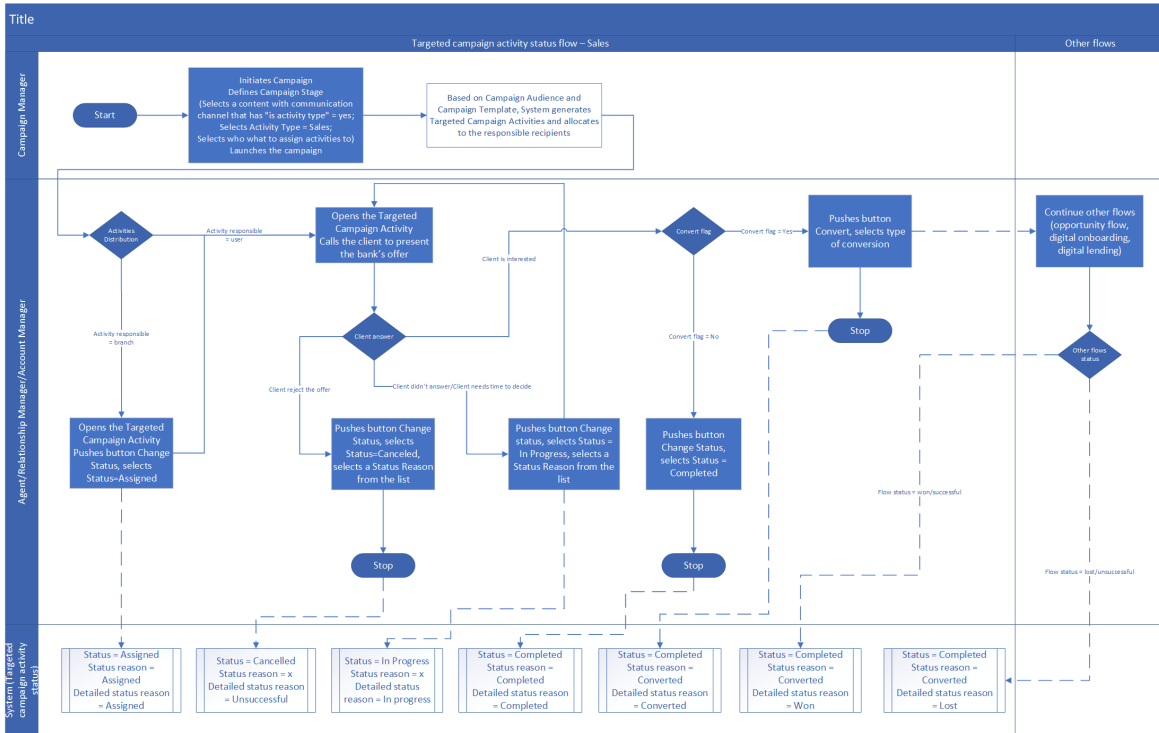
FintechOS offers the possibility of distributing targeted campaign activities more efficiently to internal users on top of the existing platform features for audiences, personas, contents, templates, actions, and activities. This feature allows bank employees to assign tasks in a practical manner and keep track of updates or to add the activity to a queue if it's necessary to contact certain clients. All of these actions are monitored at user and centralized level. Internal campaigns are needed for a better structure on the generated campaign activities. They can be [sales](#) or [administrative](#) actions that can be assigned and managed at manager or execution level.

For more details on internal campaign activities, see the following pages:

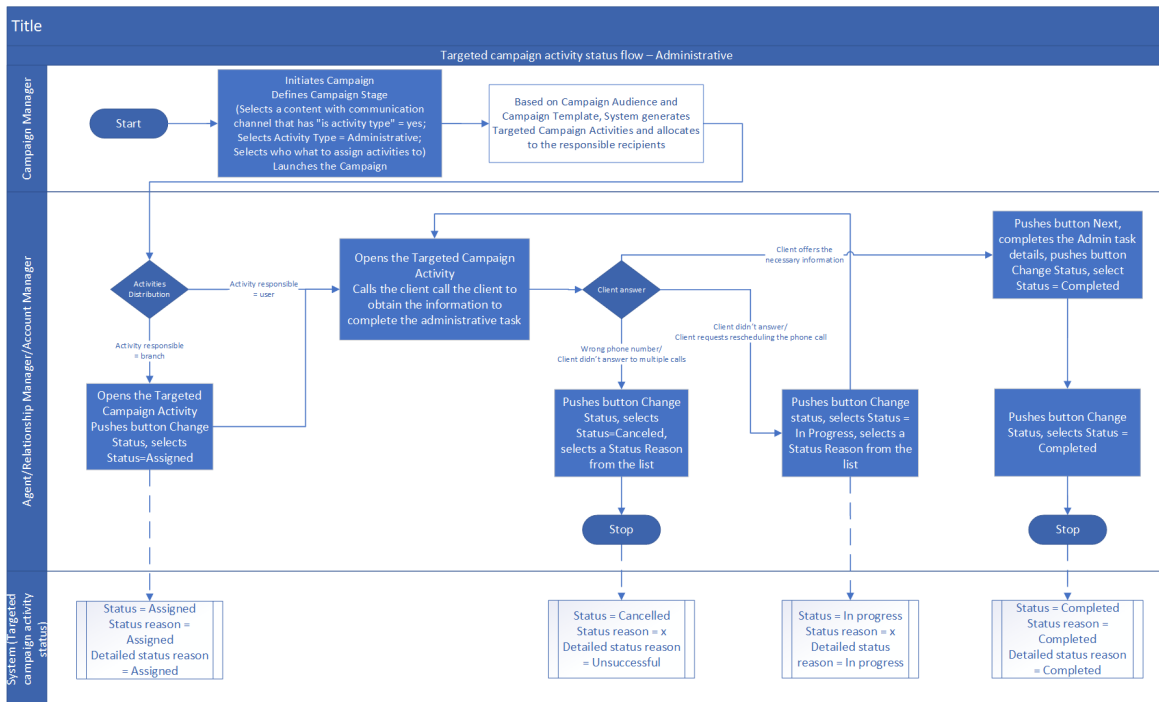
- [Internal Campaign Activities](#)
- [Dashboards](#)

The images below are UML diagrams showing the sales and administrative campaigns workflows.

AUTOMATION BLOCKS USER GUIDE



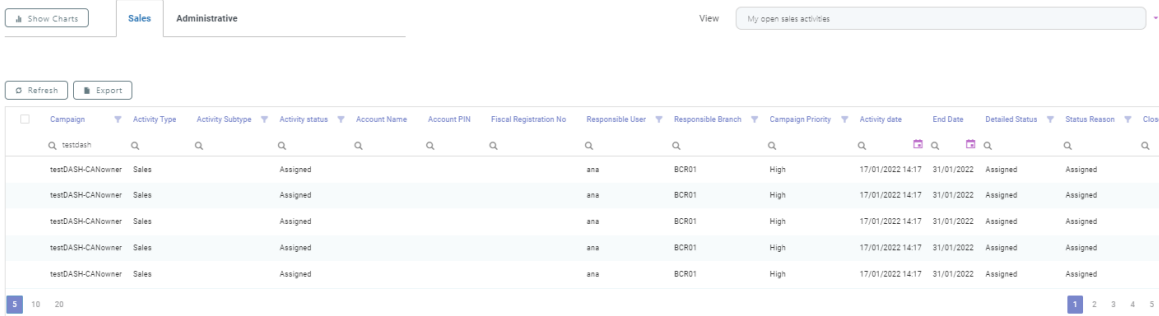
To download the Sales workflow diagram in Visio, click [here](#).



To download the Administrative workflow diagram in Visio, click [here](#).

Internal Campaign Activities

After setting the [status reasons template](#) for an internal campaign, sales or administrative campaign activities are generated. Internal campaign activities are generated after a campaign that generates internal activities is defined and approved. The person responsible for contacting the clients can access each campaign activity and based on the information presented on the activity level, the client is contacted. After the client feedback is received, the status of each campaign activity is updated. For both sales and administrative campaign types, the below data is displayed:



Field	Description
Campaign	The campaign number.
Activity Type	The activity type: sales or administrative.
Activity Subtype	The activity subtype.
Activity status	The activity status.
Account Name	The name of the account.
Account PIN	The account PIN.
Fiscal Registration No	The fiscal registration number of a corporate account.
Responsible User	The person responsible of the campaign activity.
Responsible Branch	The bank branch responsible of the campaign activity.
Campaign Priority	The campaign priority.
Activity date	The date the campaign activity begins.
End Date	The date the campaign activity ends.
Detailed Status	A detailed status of the campaign activity.
Status Reason	The status reason.
Closed On	The date the campaign was closed on.
Completed	If True , it indicates the campaign activity is completed.

HINT

The data from the activities grid can be filtered for a faster record search.

There are two types of generated internal campaign activities:

- Sales activities where bank clients are contacted for different product or services promotions.
- Administrative activities where bank clients are contacted whenever additional data or documents are needed.

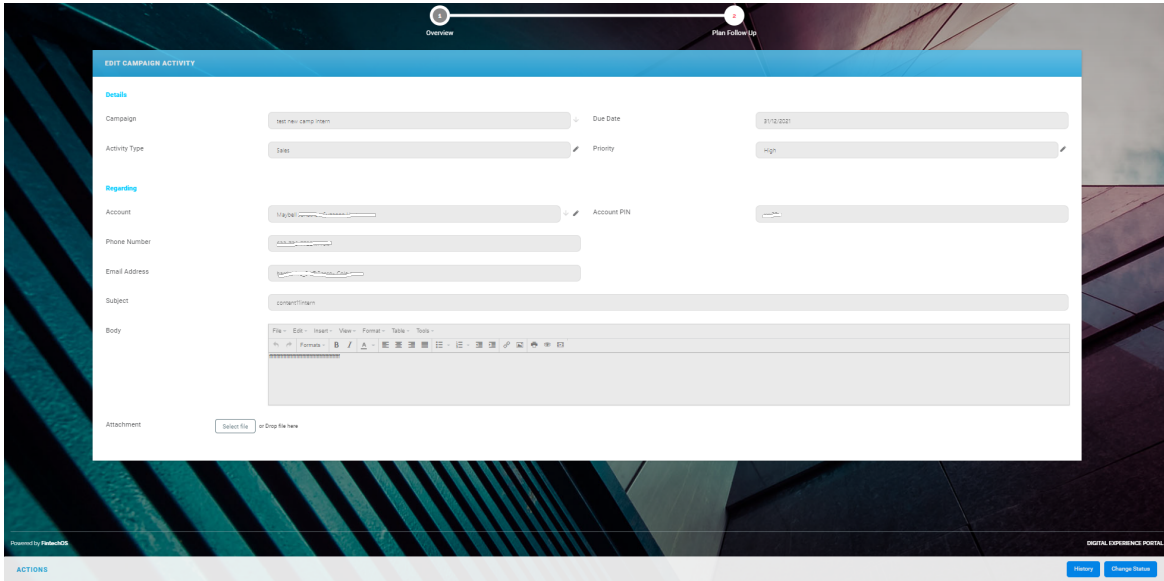
NOTE

Administrative campaign activities can have different subtypes. For example, there can be an administrative campaign with the role of collecting or updating client personal data or documents. Administrative activity subtypes are dynamic and can be set during the product implementation.

For the person that launches and manages the campaign, these activities can be viewed in the **Activities** tab and are displayed only when the campaign generates sales or administrative internal campaign activities. The person responsible with handling the campaign activities can access and view them from the **Dashboards** area.

The below steps indicate how to access additional internal campaign activities as a campaign manager in charge of launching the campaign.

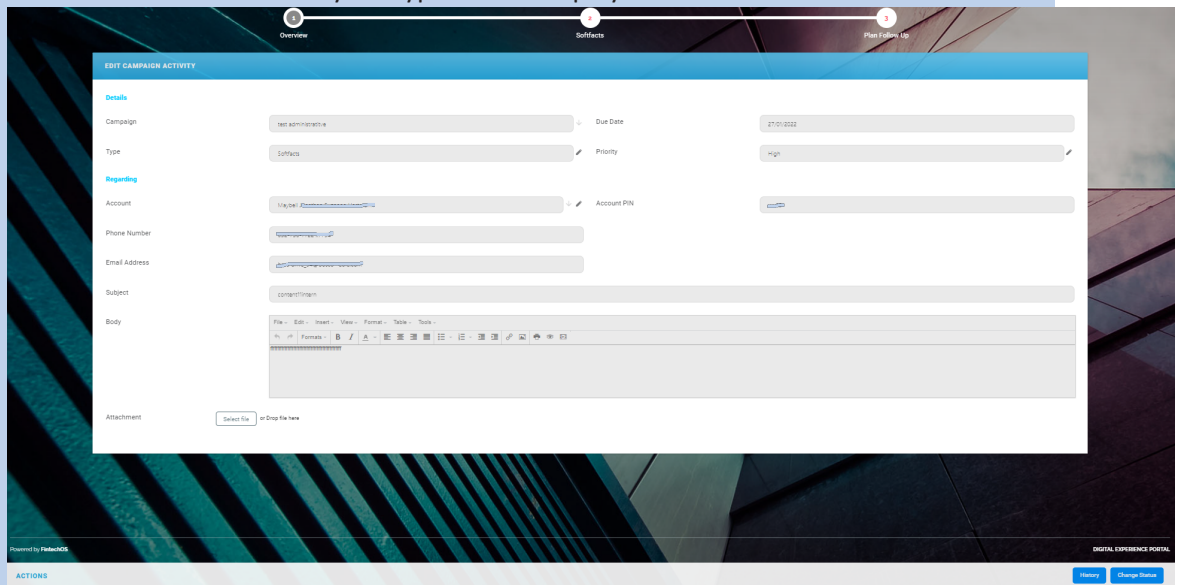
1. From the **Internal Campaign Activities** section, select a campaign activity to open it. The **Edit Campaign Activity** page is displayed.



2. The following data can be viewed:

NOTE

In case of campaigns that generate administrative campaign activity, a new tab that contains the defined activity subtype data is displayed.



AUTOMATION BLOCKS USER GUIDE

Overview Tab

EDIT CAMPAIGN ACTIVITY

Details

Campaign: test045H-CA/owner Due Date: 31/01/2022

Activity Type: Sales Priority: High

Regarding

Account: Account PIN:

Phone Number: 632-736-7722 x7794

Email Address: aahard_drive_66@8ozco-Cole.com

Subject: contact@8ozco.com

Body:

File - Edit - Insert - View - Format - Table - Tools -
 Format - B / A - [Rich Text Editor Icons]
 [Empty text area for body content]

Attachment: or Drop file here

Field	Required	Type	Description
Account	No	Read Only	The account number. If no value is set for this fields, the Account Name field value is displayed instead.
Account name	No	Read Only	The name of the account. Displayed only if the Account field is null .
Responsible User	No	Read Only	The person responsible of the campaign activity.
Responsible Branch	No	Read Only	The bank branch responsible of the campaign activity.
Status Reason	No	Read Only	The status reason. For additional information, see the Managing Reason Templates page.
Campaign	No	Read Only	The campaign number.

Field	Required	Type	Description
Due Date	No	Read Only	<p>The end date of the campaign and the date by which the activity must be closed.</p> <p>If the system date is greater than or equal to the campaign due date, all internal campaign activities are closed and the statuses are changed as follows:</p> <ul style="list-style-type: none"> • The Activity status value is changed to Cancelled • The Status reason value is change to System aborted • The Detailed status value is changed to Unsuccessful <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #000; margin-top: 10px;"> <p>NOTE If there are internal campaign activities with Activity status different than Unassigned, Assigned, or In Progress, then the campaign activities' original status is not changed.</p> </div>
Activity Type	No	Read Only	The activity type. Displayed only for Sales campaigns.
Type	No	Read Only	The administrative activity type. Displayed only for Administrative campaigns.
Priority	No	Read Only	<p>The campaign priority. The following options are available:</p> <ul style="list-style-type: none"> • High • Medium • Low • Very Low
Subject	No	Read Only	The email subject.

AUTOMATION BLOCKS USER GUIDE

Field	Required	Type	Description
Body	No	Read Only	The email body.
Phone Number	No	Read Only	The client's phone number.
Email Address	No	Read Only	The client's email address.
Account PIN	No	Read Only	The account's PIN number. Displayed only if no value is set.
Fiscal Registration Number	No	Read Only	The fiscal registration number of a corporate account.
Closed On	No	Read Only	This field is displayed only if the activity status is Cancelled or Completed . The system date value is automatically updated when the activity status is Cancelled or Completed .
Related Journey	No	Read Only	System input after converting a sales activity to an Opportunity/ Digital lending/ Digital onboarding. This field is displayed only for sales activity types.

Field	Required	Type	Description
Current Status	No	Read Only	<p>The current status of the campaign. The following options are available:</p> <ul style="list-style-type: none"> • Unassigned: set automatically by the system when the campaign is started and the activities are distributed to the responsible branch • Assigned: set automatically by the system when the campaign is started and the activities are distributed to a user (account responsible/ a user from the responsible branch) <div data-bbox="836 840 1369 1123" style="background-color: #e6f2ff; padding: 10px; border: 1px solid #ccc;"> <p>NOTE When a user changes the status from Unassigned to Assigned, the responsible user changes automatically with the person who changed the status.</p> </div> <p>These statuses can be set manually by the user:</p> <ul style="list-style-type: none"> • Cancelled • In progress • Completed

Field	Required	Type	Description
Detailed Status	No	Read Only	<p>A detailed status of the campaign activity.</p> <ul style="list-style-type: none"> • If Activity Status is Unassigned, then Detailed status is Unassigned • If Activity Status is Assigned, then Detailed status is Assigned • If Activity Status is In progress, then Detailed status is In progress • If Activity Status is Cancelled, then Detailed status is Unsuccessful • If Activity Status is Completed and Related Journey is null and Opportunity status is Open, then Detailed status is Completed • If Activity Status is Completed and Related Journey is not null and Opportunity status is Won, then Detailed status is Won • If Activity Status is Completed and Related Journey is not null and Opportunity status is Lost, then Detailed status is Lost
Add file	No	File	Allows the upload of necessary documents.

At the bottom of the page the following action buttons are displayed: **Change Status**, **History**, **Assign to user**, **Assign to me**. For additional details, see the [Action buttons](#) page.

Plan Follow Up Tab

In this tab, meeting and call follow-up activity types are displayed.

IMPORTANT!

When a campaign activity is either in completed or cancelled status, the following fields and buttons become read-only:

- all the fields from the **Activity** section of the **Overview** tab
- all the following buttons: **Change Status** and the standard platform buttons as well (**Save & Close**, **Save & Reload**, etc.)

Dashboards

Automation Blocks offers the possibility of viewing centralized campaign activities at execution and manager level. This feature aids campaign managers assign campaign tasks in a practical manner and keep track of updates. In addition, execution users can view their assigned activities.

To access the dashboard, log into the FintechOS Portal with the given credentials and select the **Internal Campaigns** tab. In this page, the following tabs are displayed:

NOTE

The activities displayed are based on the user role. For example, a manager can view the activities related to their team, while a user with an execution role views their activities or their branch's unassigned activities.

Sales Activities

In this tab, all relevant information related to an internal sales activity is displayed. This allows users to contact retail or corporate clients in order to promote company products or services. Here, the following tasks can be viewed:

- **My sales activities/ My team's sales activities:**
 1. The open sales activities option displays the active sales campaign activities list that have the following statuses: **Assigned** or **In progress**.
 2. The closed sales activities option displays the sales campaign activities list that have the following statuses: **Completed**, **Converted**, **Won**, **Lost**, or **Unsuccessful**.
- **My branch unassigned activities/ My team's unassigned activities** displays the activities that are not allocated to a user. Here, users can assign tasks based on their user role using the [Assign to me](#) (execution view) or [Assign to user](#) (manager view) buttons.

Administrative Activities

In this tab, all the relevant information related to an internal administrative activity is displayed in order to help the person in charge of the campaign understand and finalize the campaign activity. Here, the following tasks can be viewed:

- **My administrative activities/ My team's administrative activities:**
 1. The open administrative activities option displays the active administrative campaign activities list that have the following statuses: **Assigned** or **In progress**.
 2. The closed administrative activities option displays the administrative campaign activities list that have the following statuses: **Completed** or **Unsuccessful**.

- **My branch unassigned activities/ My team's unassigned administrative activities** displays the activities that are not allocated to a user. Here, users can assign tasks based on their user role using the [Assign to me](#) (execution view) or [Assign to user](#) (manager view) buttons.

NOTE

For analysis and reporting purposes select the desired records from the campaign list and click the **Export** button to download the data set. If the grid is filtered or truncated (displaying 5, 10, 20, etc. entries per page out of a larger set), the current set export includes only the displayed values.

On both tabs the following data is displayed:

Field	Description
Campaign	The campaign number.
Activity Type	The activity type: sales or administrative.
Activity Subtype	The activity subtype.
Activity status	The activity status.
Account Name	The name of the account.
Account PIN	The account PIN.
Fiscal Registration No	The fiscal registration number of a corporate account.
Responsible User	The person responsible of the campaign activity.
Responsible Branch	The bank branch responsible of the campaign activity.
Campaign Priority	The campaign priority.
Activity date	The date the campaign activity begins.
End Date	The date the campaign activity ends.
Detailed Status	A detailed status of the campaign activity.
Status Reason	The status reason.

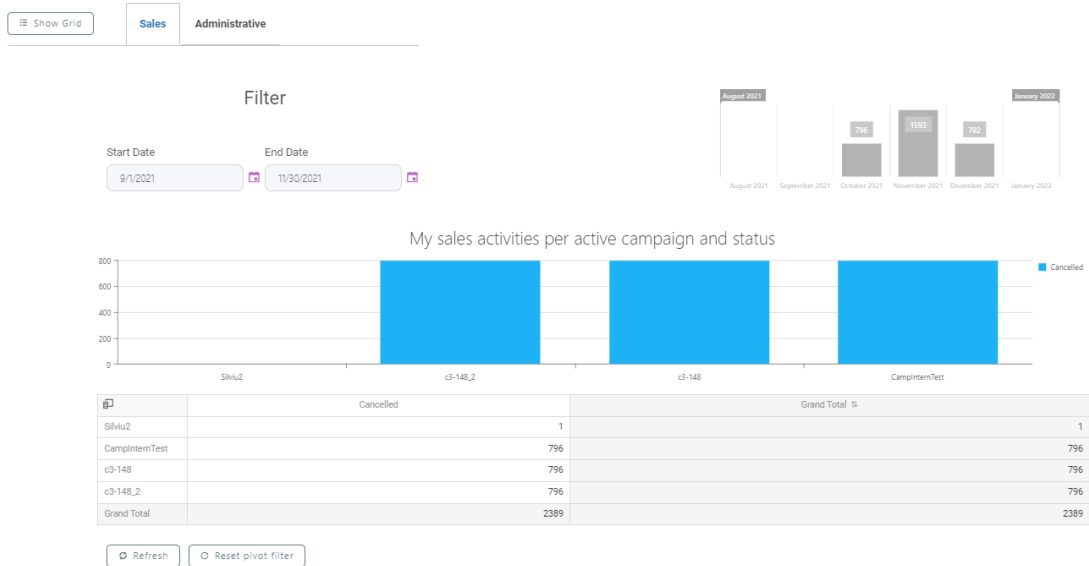
Field	Description
Closed On	The date the campaign was closed on.
Completed	If True , it indicates the campaign activity is completed.

HINT
The data from the activities grid can be filtered for a faster record search.

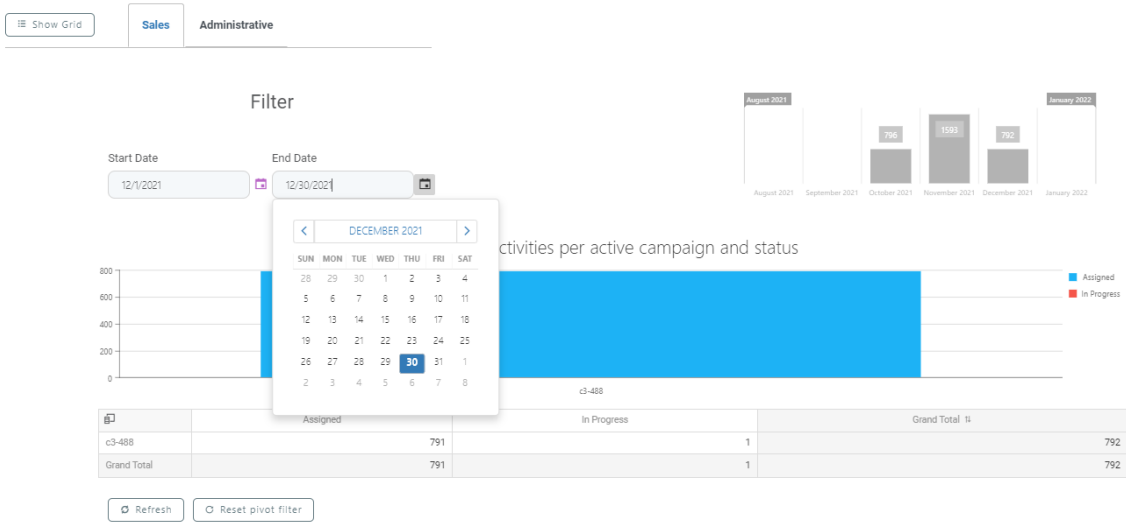
Viewing Activities Dashboard Charts

Dashboard charts are useful tools that can be used to identify which activities are assigned to a certain bank branch or employee to easily see the status of each task.

To view the charts, select either the **Sales** or **Administrative** tab and then click the **Show Charts** button from the upper left corner. The displayed is filtered by default for the last three months.



To filter the display period , change the **Start Date** and the **End Date**.



Filtering Dashboard Data

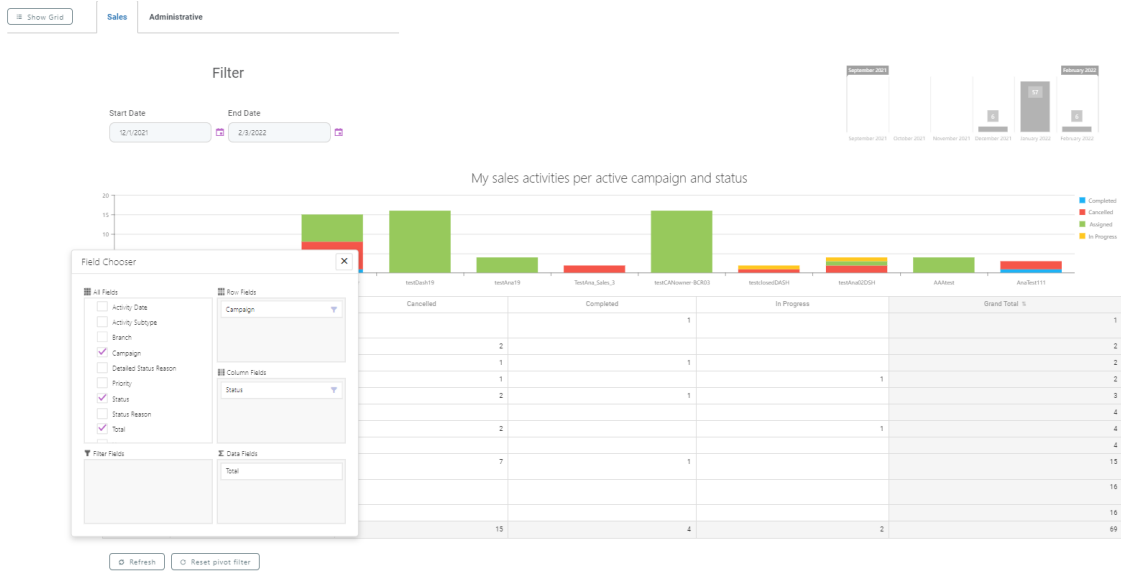
To change the way the information from the pivot table is viewed, select the **Show Field Chooser** option. A pivot filter is displayed with the following default settings:

- The row fields display the **Campaign, Branch, and User** fields.

NOTE

The default settings are different depending on the user's role. In case of a user with execution role, the **Branch** and **User** fields are not displayed.

- The column fields display the **Status** field.
- The data fields display the **Total** field.



The pivot option is customizable depending on the data needed. To return to the default settings, click the **Reset pivot filter** button.

The **Campaign/ Branch/ User, Status, and Total** fields are displayed based on the rules set in the pivot table option. For example, each manager sees only the activities assigned to direct or indirect subordinates and the activities assigned to their branch or to the branch that is a direct or indirect child of their branch. The same rules apply to the person the campaign activity is assigned to. They are able to view only the activities assigned to them.

HINT
The pivot settings are saved for each particular user and are not reset after logging off.

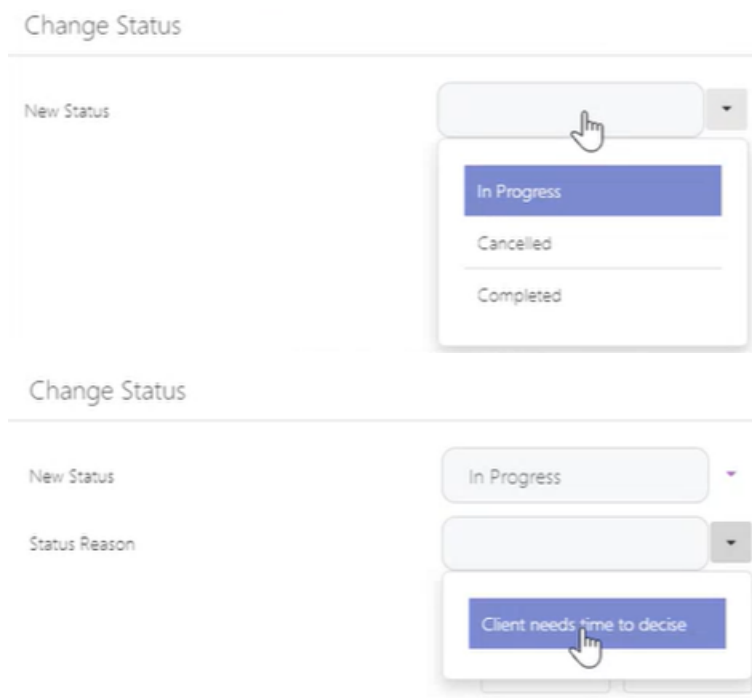
To return to the grid view, click the **Show Grid** button from the upper left corner.

Actions Buttons

Action buttons are meant to help the user better manage the internal campaign activities generated. The following actions buttons are displayed at the bottom of the screen when handling internal campaign activities:



- **Change Status:** allows the user to change the status of the campaign to: **In Progress**, **Cancelled**, or **Completed**.



When changing the status reason, only the reasons from the template set up at stage setup level can be selected.

- **History:** displays a tracking history of the actions taken on the campaign for the following fields:
 - Activity Status
 - Status Reason
 - Detailed Status
 - Activity Owner (User)
 - Responsible Branch

HISTORY					
Date	Modified by	Changes			
		Field	Old value	New value	
* Month: December -2021					
12/13/2021 8:21:38 PM	Teodora2	Detailed Status Status Reason	Assigned Assigned	Unsuccessful Client Refused Offer	
12/13/2021 8:21:38 PM	Teodora2	Activity status	Assigned	Cancelled	

[Close](#)

- **Assign to user:** used by users with manager role to select a user to assign campaign activities to. Only campaign activities that have **Unassigned, Assigned, or In progress** activity status can be assigned.

The manager allocates the unassigned activities, in bulk or one at a time, to one of the users that are part of the campaign activity responsible branch or between users from the same branch.

HINT

This option is available either from the [Dashboard](#) view or at [campaign activity](#) level.

If a manager tries to assign campaign activities to a user that is not part of the campaign activity responsible branch, the following message is displayed: You are not allowed to assign campaign activities to a user from a different branched.

If a manager tries to assign campaign activities between users from different branches the following message is displayed: You are not allowed to assign campaign activities between user from different branched.

In the case of a bulk assign, if only some of the activities meet the assign conditions, the activities that meet the assign conditions are reallocated and the following message is displayed: Campaign activities that meet the reallocation conditions have been reallocated. The following campaign activities did not meet the assignment conditions:

- **Assign to me:** used by users with execution role to self assign, in bulk or one at a time, campaign activities that are assigned to the branch that they are part of and are unassigned.

HINT

This option is available either from the [Dashboard](#) view or at [campaign activity](#) level.

IMPORTANT!

For both **Assign to user** and **Assign to me** buttons, there is a possibility to display or hide these options depending on the user role and the activity status. These conditions are disabled by the technical team at the time of implementation.

Creating Multi-Stage Execution Plans

Campaign managers have the possibility of creating a multi-stage execution plan for the campaign stages that can be started manually, set to run automatically at a certain date and time, or with a defined recurrence. To set up a multi-stage execution plan, in the Innovation Studio main menu, go to **Automation Blocks > Omnichannel Campaigns > Multi-Stage Execution Plan**.

IMPORTANT!

Multi-stage execution plans can be set up only if at stage level the stages included in the execution plan have the **Controlled By Execution Plan** field set to true and the business status of the campaign is **Approved** with the closing date set in the future.

There are two ways to start an execution plan:

- Manually, by using the **Start Now** button after setting the transition and schedule type.
- Automatically, by defining the schedule rules for a date set in the future, or with a certain recurrence.

Follow the below steps to define and run a multi-stage execution plan.

1. In the **Multi-Stage Execution Plan List**, click the **Insert** button from the right upper corner. The **Multi-Stage Execution Plan** page is displayed.

2. Fill in the fields:

Multi Stage Execution Plan

Name

Stage Transition - ✎

Schedule Type - ✎

Scheduled Date 🗓

Scheduled Timezone - ✎

STAGES AVAILABLE

Stage	Campaign	Campaign Code
<input type="checkbox"/> demo	<input type="text" value="demo"/>	
<input checked="" type="checkbox"/> StageDemo1	CampaignDemo1	
<input checked="" type="checkbox"/> demo10	demo10	

EXECUTION TREE

Stage	Campaign
<input checked="" type="checkbox"/> StageDemo1	CampaignDemo1
<input checked="" type="checkbox"/> demo10	demo10

MULTI-STAGE EXECUTION PLAN INSTANCES

Name	Created On	Finished At	Execution Status
<input type="text" value="demo2"/>	<input type="text" value="demo2"/>	<input type="text" value="demo2"/>	<input type="text" value="demo2"/>
demo2 - 2022-03-22 05:49	22/03/2022 16:49	22/03/2022 16:50	Completed
demo2 - 2022-03-22 05:30	22/03/2022 17:30	23/03/2022 18:51	Cancelled

Field	Required	Type	Description
Name	No	Text	The name of the execution plan.

Field	Required	Type	Description
Stage Transition	Yes	Option Set	<p>Establish when to move from one stage to another. This setting applies only when two stages are codependent and it determines when the second stage begins. The available options are:</p> <ul style="list-style-type: none"> • When Stage Completed: the transition occurs when all the messages have been created but have not been yet sent to the recipients. • When Stage Sent: the transition occurs when all the generated communications from the message queue are sent to the recipients.

Field	Required	Type	Description
Schedule Type	Yes	Option Set	<p>Schedule when to run the execution plan. The available options are:</p> <ul style="list-style-type: none"> • Manual: The execution plan starts when the Start Now button is pressed. • One Time: The execution plan is scheduled to start at a date and time set in the future but only if the execution plan has an Approved business status. • Recurrent: Define a recurrence for when new multi-stage execution plan instances are created.

Field	Required	Type	Description
Scheduled Date	Yes	Date	<p>Schedule the date and time when to run the execution plan.</p> <div style="background-color: #e1ecf4; padding: 10px; border-radius: 5px;"> <p>NOTE This field is available only if the Schedule Type field is set to One Time.</p> </div>
Cron Expression	No		<p>Schedule a recurrence for the execution plan. For example, the execution plan can be scheduled to run every day at 9 AM.</p> <div style="background-color: #e1ecf4; padding: 10px; border-radius: 5px;"> <p>NOTE This field is available only if the Schedule Type field is set to Recurrent.</p> </div>

Field	Required	Type	Description
Scheduled Timezone	Yes	Option Set	<p>Select the timezone that applies to the execution plan scheduled date.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 10px; margin-top: 10px;"> <p>NOTE This field is available only if the Schedule Type field is set to either Recurrent or One Time.</p> </div>

Stages Available

This section displays the stages that can be used when running the execution plan.

Field	Description
Campaign	The campaign name that is part of the stage.
Campaign Code	The campaign code.
Stage	The name of the stage.

Execution Tree

This section displays the stages selected when running the execution plan. By default, the execution plan runs the stages from the execution tree from top to bottom. Stages can be sequentially ordered by using the drag and drop option to place one

stage after another.

STAGES AVAILABLE

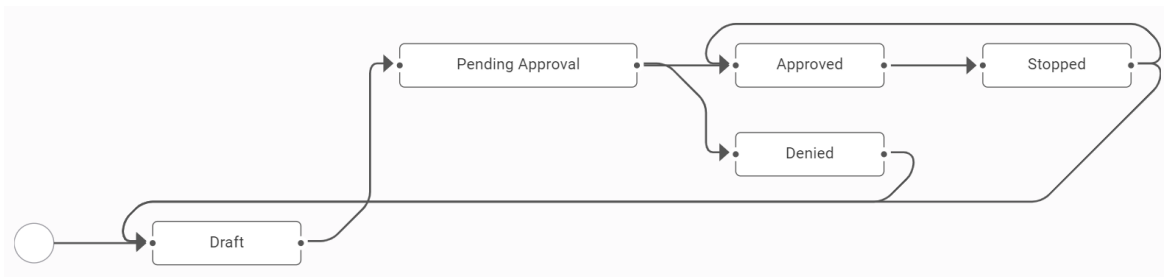
Campaign	Campaign Code	Stage
<input type="text" value="demo"/>		<input type="text" value=""/>
<input type="checkbox"/>	CampaignDemo1	StageDemo1
<input checked="" type="checkbox"/>	DemoCampaign2	Demo2Stage2
<input checked="" type="checkbox"/>	DemoCampaign1	Demo1Stage2
<input checked="" type="checkbox"/>	DemoCampaign2	Demo2Stage1
<input checked="" type="checkbox"/>	DemoCampaign1	Demo1Stage1

EXECUTION TREE

Stage	Campaign
▼ Demo1Stage2	DemoCampaign1
▢ Demo2Stage2	DemoCampaign2
▢ Demo1Stage1	DemoCampaign1
▢ Demo2Stage1	DemoCampaign2

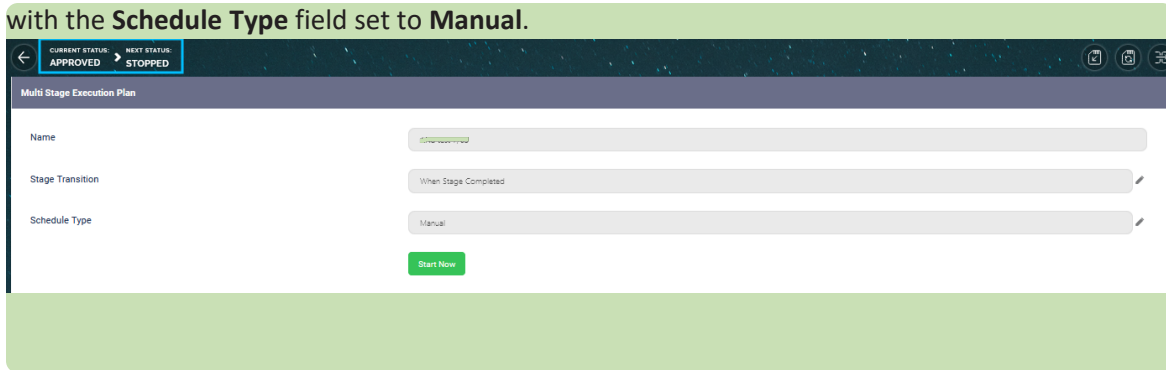
Field	Description
Stage	The name of the stage.
Campaign	The name of the campaign.

After the schedule rules are defined, the campaign manager can set different business statuses for the execution plan based on the business workflow transitions. The execution plan can have the following statuses: **Draft**, **Pending Approval**, **Approved**, **Stopped**, **Denied**.



When the execution plan is in **Draft** status, changes can be made to it by campaign managers. From the **Pending Approval** status, it can be either approved or denied. If it's in **Approved** status, then the execution plan can be run, creating new execution plan instances.

HINT
 The **Start Now** button is functional only for execution plans in **Approved** statuses



If the status is changed to **Stopped**, then another execution plan instance does not start running on that execution plan. In case an execution plan instance is running while the campaign manager changes the status to **Stopped**, then the execution plan instance runs until the end.

After clicking the **Start Now** button, campaign managers can pause, resume, or cancel the delivery of the messages generated by an execution plan instance by using the following options:

- **Pause delivery:** Use this button to stop the delivery of all the remaining messages that have not been sent to the recipient on that stage instance. In this case, messages are still created but are no longer sent to the recipients.

This button is displayed if an execution plan instance is in progress.

- **Resume delivery:** Use this button to resume the delivery of the messages that haven't been sent to the recipient.

This button is displayed if an execution plan instance is paused.

To stop the instance execution process while the execution plan is running, click on the **Stop current instance execution** button. This button is visible both at multi-stage execution plan and multi-stage plan instance level.

When clicking the **Stop current instance execution** button at execution plan level, the stage that is currently running stops. The process of creating and delivering messages is cancelled and the messages that remain to be sent to the recipients are no longer sent.

The **Stop current instance execution** button that stops running the multi-stage execution plan instance, can be used regardless of the multi-stage execution plan status.

Multi Stage Execution Plan

Name

Stage Transition

Schedule Type

Stop current instance execution

After the multi-stage execution plan starts running, a summary showing the statuses of the execution stage instances that are running is displayed.

Multi Stage Execution Plan

Name

Stage Transition

Schedule Type

Stop current instance execution

Campaign Name	Campaign Code	Stage Name	Status	Start Date	End Date
- DocTest - 2022-03-25 06:38			Started	3/25/2022, 8:38 AM	
MultiPlanTest1		MPTStage2	Completed	3/25/2022, 8:38 AM	3/25/2022, 8:38 AM
MultiPlanTestStaticAudience		Stage123	Running	3/25/2022, 8:38 AM	

MULTI-STAGE EXECUTION PLAN INSTANCES

Refresh

Name	Created On	Finished At	Execution Status
DocTest - 2022-03-10 04:00	10/03/2022 18:00		Completed
DocTest - 2022-03-10 04:00	10/03/2022 18:00	11/03/2022 17:00	Completed
DocTest - 2022-03-14 12:42	14/03/2022 14:42	14/03/2022 14:44	Cancelled
DocTest - 2022-03-25 06:35	25/03/2022 08:35	25/03/2022 08:38	Completed
DocTest - 2022-03-25 06:36	25/03/2022 08:36	25/03/2022 08:38	Cancelled
DocTest - 2022-03-25 06:38	25/03/2022 08:38		Running

Use the **Refresh** button to manually update the running statuses of the execution plan.

NOTE
The execution plan form is automatically updated once the execution plan starts running and the execution plan instance is created.

Field	Description
Campaign Name	The name of the stage.
Campaign Code	The code of the campaign.
Stage Name	The name of the stage.

Field	Description
Status	The status of the stage: <ul style="list-style-type: none"> • Not Started • Started • Running • Completed • Cancelled
Start Date	The start date of the multi-stage execution plan instance.
End Date	The end date of the multi-stage execution plan instance.

Multi-Stage Execution Plan Instances

This section displays the history log of every execution plan run. If, for example, the execution plan has a scheduled recurrence, then every recurrence log is displayed.

Double-click on a record for additional details. A summary of the execution plan is showed:

AUTOMATION BLOCKS USER GUIDE

MULTI-STAGE EXECUTION PLAN INSTANCE

Name	Demo5 - 2022-02-28 01:39	Parent Execution Plan	Demo5
Execution Status	Completed	Finished At	28/02/2022 15:41

Execution Tree

Stage	Campaign
▼ Demo1Stage2	DemoCampaign1
Demo2Stage2	DemoCampaign2
Demo1Stage1	DemoCampaign1
Demo2Stage1	DemoCampaign2

EXECUTION LOG

Date	Time	Event	Stage	Campaign	Details
2/28/2022	3:40:15 PM	Stage Instance Executed	Demo1Stage2	DemoCampaign1	Stage Instance executed in 0:00:00.874223 (dd:hh:mm:ss).
2/28/2022	3:40:00 PM	Stage Instance Created	Demo1Stage2	DemoCampaign1	
2/28/2022	3:40:00 PM	Stage Instance Created	Demo2Stage1	DemoCampaign2	
2/28/2022	3:40:00 PM	Stage Instance Created	Demo1Stage1	DemoCampaign1	
2/28/2022	3:39:45 PM	Plan Started			Triggered start manually on demand

5 10 20 1 2 3

CAMPAIGN STAGE INSTANCES

[Export](#) [Refresh](#)

<input type="checkbox"/>	Campaign	Stage	Start Date	Activity Date	Completed	End Date	Activity Cre...	Activity Fini...	Has Errors	Run Log
<input type="checkbox"/>	DemoCamp...	Demo2Stag...	28/02/2022...		<input checked="" type="checkbox"/>	28/02/2022...			<input type="checkbox"/>	Stage comp...
<input type="checkbox"/>	DemoCamp...	Demo1Stag...	28/02/2022...		<input checked="" type="checkbox"/>	28/02/2022...			<input type="checkbox"/>	Stage comp...
<input type="checkbox"/>	DemoCamp...	Demo2Stag...	28/02/2022...		<input checked="" type="checkbox"/>	28/02/2022...			<input type="checkbox"/>	Stage comp...
<input type="checkbox"/>	DemoCamp...	Demo1Stag...	28/02/2022...		<input checked="" type="checkbox"/>	28/02/2022...			<input type="checkbox"/>	Stage comp...

Field	Description
Name	The name of the multi-stage execution plan instance composed from the multi-stage execution plan name and and the time the instance began to run.

Field	Description
Parent Execution Plan	The parent multi-stage execution plan name.
Execution Status	The multi-stage execution plan instance status.
Finished At	The date and time the multi-execution plan instance finished running.
Execution Tree	A snapshot of the execution tree.

Execution Log

This section displays the actions taken along with the associated stage and ending dates.

Field	Description
Date	The date the execution plan started.
Time	The time the execution plan started
Event	The stage instances events created. For example: Stage Instance Created, Activities/ Actions Created, Queue Items Created, etc.
Stage	Hyperlink to the stage.
Campaign	Hyperlink to the campaign.
Details	Details regarding the execution plan.

Campaign Stage Instances

This section displays information regarding the campaign stage instances. The below data is available.

Field	Description
Campaign	The name of the stage.
Stage	The code of the campaign.
Start Date	The start date of the stage instance.
Activity Date	The date of the campaign activity.
Completed	If true, the stage is completed.
End Date	The end date of the execution plan.
Activity Creation Date	The date the activity is created.
Activity Finish Date	The date the activity is finished.
Has Errors	If true, errors occurred when running the execution plan.
Run Log	The run log of the execution plan.

HINT

To view information regarding campaign stage instances or to pause, resume, or

cancel the delivery of the messages generated by a stage instance, select a record from the list and open it. For more details, see the [Campaign Stage Instances \(Actual Run\)](#) section.

Omnichannel Communication Automation

The Omnichannel Communication Automation enables you to overcome the challenge of email delivery.

This automation processor allows you to use the FintechOS Gateway as email server and send emails on your company's behalf. You can handle email delivery and track real-time email events directly from within the FintechOS platform. The email communication with your customers is reliably sent and delivered as needed.

Features

- Easy configuration
- Open Email & Click URLTracking
- Email Template Engine.

Installation

Innovation Studio comes with the Omnichannel Communication Provider processor pre-installed.

Applications

- Omnichannel campaigns for notifying customers.

Sendinblue Email Provider

Sendinblue is an email provider which you can integrate with FintechOS to create and send transactional emails, such as newsletters and release announcements, "triggered" emails such as order confirmations, password resets, confirmation messages, and more.

Models

Multi Account:

- used with shared IP for each sub-account
- there will be a master account that give the possibility to create the sub-accounts. Practically, the Master Account(1 fintechOs account) is used to create sub-accounts(for each fintechOS' client). Each sub-accounts is a completely independent entity with fully rights.

Configuration

Get API Key

Each sub-account has an API KEY. It is possible to create as many API KEY as we want on each sub-account, without limitation. FintechOS configures 3 API-Key per each sub-account (TEST, UAT, PRD).

Domain and IP

Sub-accounts can be configured on shared IPs. This is configured by CSM team based on performances. The IP is configured and pointed it to the domain and then configured the sender.

HINT

If needed, a subdomain can be added to the Sendinblue functionality. For example:

- sender address: hello@**example**.ex.org
- domain signature (DKIM): **example**.ex.org

Each sub-account can have as many senders as need as long as they belong to the defined domain. In case of share IP, the sub-account does not need to be pointed to the domain.

Sub-accounts are created from the master account, each sub-account can have different plan.

Capabilities

The dashboard for monitoring the service displays the transactional activities available in the corresponding account.

The status of sending and the logs are available real time. You are always in control of the reporting by being able to filter per logs, tags (transactional), sender (transactional), as well as set limits per account.

Transactional

Available via interface(UI) and API.

Statistics can filter per sender, tag, and are displays all the logs. Each log has all the history of the e-mails. It is not possible to filter by API key.

Webhooks

Webhooks are sent by Sendinblue each time an event occurs on the email message sent to the client. These events can be:


- **request** - Triggered once the e-mail is sent
- **click** - Triggered when user clicks in the e-mail
- **opened** - Triggered each time the user opens the e-mail
- **unique_opened** - Triggered when the user opens the e-mail for the first time
- **delivered** - Triggered when the e-mail is received by the user

- `deferred`
- `soft_bounce` -> If there was something wrong sending the e-mail. Will retry 3 times.
- `complaint`
- `hard_bounced`
- `invalid_email`
- `blocked`
- `error`
- `unsubscribed`

These statuses, alongside all the original request data can be found in the `Url` attribute in `MessageEvent` list for the given Message Queue. In order to update the status of the message in `FTOS_DPA_MessageQueue`, we create a mapping to the existing statuses in Fintech.

Create the Channel Configurations

1. Log into the Innovation Studio using a developer account.
2. From the menu, click **Admin > Omnichannel Communication Automation > Channel Configurations**.
3. The page Channel Configurations List is displayed. Click on the **Insert** button to add a new channel configuration.

CHANNEL CONFIGURATIONS LIST	
<input type="checkbox"/>	Name
	
	AzureNotificationHub
	Email
	Sms

The Add Channel Configurations page appears which allows you to create a new configuration.

4. Fill in the following fields:

EDIT CHANNEL CONFIGURATION	
Name	<input type="text" value="Email"/>
code	<input type="text" value="2"/>

Field	Data type	Description
Name	Text	Insert a suggestive name.
Code	Whole number	Insert a unique code.

5. At the top-right corner of the page, click the **Save and reload** icon to save the changes.
The Edit Channel Configurations page appears.

Configure Omnichannel Communication Channel Providers

1 Apply for subscription key

Go to the FintechOS Services Portal and apply for a subscription key for the FintechOS Omnichannel Communication Automation processor. You need the subscription key to configure the automation processor, which is editing the FTOSEmailGateway channel provider so that it uses this processor.

2 Edit the FTOSEmailGateway channel provider

To configure the Automation Blocks you need to edit the FTOSEmailGateway channel provider:

1. Log into the Innovation Studio using a developer account.
2. From the menu, click **Admin > Omnichannel Communication Automation > Channel Providers**.
3. Double click on the 'FTOSEmailGateway' record.

CHANNEL PROVIDERS LIST	
Name	Communication Channel
FTOSApSms	Sms
<input checked="" type="checkbox"/> FTOSEmailGateway	Email
GatewayEmail	Email
GatewayEmailOTP	Email
GatewaySms	Sms
GatewaySmsOTP	Sms

The Edit Channel Provider page appears which allows you to update settings and also see the list of [channel provider statuses](#).

4. Fill in the following mandatory fields:

Field	Data type	Description
Communication Channel	Option set	Select email from the option set. For details, see "Create the Channel Configurations" on page 369.
Name	Text	Insert the name for the gateway.
Provider Name	Text	Insert the name of the provider.
Service Url	Text	The URL provided by FintechOS
App user	Text	It is the user credentials given by the provider.
App Password	Text	It is the password given by the provider.
App Key	Text	The subscription key provided by the FintechOS Services Portal for the FintechOS Omnichannel Communication Automation processor.
appConfig	Text	The configuration file.

Field	Data type	Description
From Address	Text	<p>The email address from which the emails will be sent to customers. It is the email address displayed in the From field of the email sent to customers.</p> <div style="background-color: #f4a460; padding: 10px; border-radius: 5px;"> <p>IMPORTANT!</p> <p>Please, be sure to consult FintechOS to create an email address with the name of the company.</p> </div>
Bulk No	Whole Number	The number of emails that are processed in one step.
Interval Min	Text	It is the limit for sending messages. The job will send messages only in this time frame (HH:MM:SS) (UTC time).
Interval Max	Text	It is the limit for sending messages. The job will send messages only in this time frame (HH:MM:SS) (UTC time).
Retry Max Attempt	Whole number	The maximum number of times the server tries sending the message if the message status is 'Error'.
Pool Time Retry	Text	<p>The time interval the server waits until it attempts to send the message that has previously failed (message status is 'Error').</p> <p>The format is HH:MM:SS (hours:minutes:seconds).</p>

Field	Data type	Description
Open Setting	Bool	Tracks the open email events.
Click Setting	Bool	Tracks the clicked links within the email body events.

- At the top-right corner of the page, click the **Save and close** icon to save the changes.

Use the Omnichannel Communication Channels

The available channel provers are for e-mail and SMS. The available list in the Innovation Studio for e-mail is:

- FTOSEmailGateway
- GatewayEmail
- GatewayEmailOTP.

The available list in the Innovation Studio for SMS is:

- GatewaySMS
- GatewaySMSOTP
- FTOSApiSMS.

Email provider configurations

1 Add Communication Channel

- From the main menu, click **Admin > Omnichannel Communication Automation > Communication Channels**. The Communication Channels List page appears.
- Provide a **Name** for the communication channel.
- From the **Default Action Type** field, select **Email**.

4. From the **Bus Communication Channel** field, select **Email**. For Bus Communication Channel, see details "[Create the Channel Configurations](#)" on page 369.
5. From the **Bus Communication Provider** field, select **FTOSEmailProvider**. For details, see [How to Configure Omnichannel Communication Channel Providers](#).
6. From the **Extended Properties Entity**, select from the list the entity. Through certain channels, it allows sending some properties with the JSON format, using this channel as API.
7. At the top-right corner of the page, click the **Save and close** icon to save the communication channel.

You can now add messages to the Message queue and send them using the Omnichannel Communication Automation processor.

2 Add message

You can add messages to the FintechOS email server gateway to be processed within the specified time interval.

1. Go to the Message Queues List page ([#/entity/FTOS_DPA_MessageQueue/list](#)).
2. At the top-right corner of the page, click the **Insert** icon. The Add Message Queue page appears.
3. Provide the **ToAddress**, that is, the email address where the email should be sent.
4. Provide the **CC**, that is, one or more carbon copy email addresses separated by commas.
5. Provide the **BCC**, that is, one or more blind carbon copy email addresses separated by commas.
6. Fill in the email **Subject** and **Body**.
7. Optionally, add an attachment to the email.
8. From the **Communication Channel** field, select **Email**.

9. From the **Channel Provider** field, select **GatewayEmail**.
10. Set the interval (date and time) when you want the message to be sent.
11. Select the **Start Date**, **End Date**, **Interval Min (Time UTC)** and **Interval Max (Time UTC)** (for example: 00:00:00, 23:00:00).
12. From the **Message Status** field, select **New**.
13. From the **Channel Provider Status** field, select **200**.
14. At the top-right corner of the page, click the **Save and close** icon to save the message queue.

When the message is sent (for example, when running omnichannel campaigns), if you have configured the Omnichannel Communication Automation processor to track open emails and URL clicks, the Message Events section (Edit Message Queue Page), lists all open and click events. For URL click events, it also displays the URL which has been clicked.

SMS provider configurations

Add Communication Channel

1. From the main menu, click **Admin > Omnichannel Communication Automation > Communication Channels**. The Communication Channels List page appears.
2. Provide a **Name** for the communication channel.
3. From the **Default Action Type** and **Bus Communication Channel** fields, select **SMS**.
4. From the **Bus Communication Provider** field, select **FTOSApiSMS**.
5. At the top-right corner of the page, click the **Save and close** icon to save the communication channel.

You can now add messages to the Message queue and send them using the Omnichannel Communication Automation processor.

For details on how to implement a new provider for sending SMS through FTOS service, see [Configure the FTOSApiSMS provider](#).

For details regarding the SMS-based Two-Factor Authentication, see [SMS-based Two-Factor Authentication](#).

Channel Provider Statuses

Failed requests return an error that contains: a response code, a message explaining the reason for the error and a link to any relevant documentation that might help you troubleshoot the problem.

The table below describes the list of predefined errors but you can add others if you need.

EDIT CHANNEL PROVIDER STATUS

Code	400
Description	BAD REQUEST
Channel Provider	FTOSEmailGateway ↓ ↗
messageStatusId	Error ↓ ↗

Error Code	Error reason	Description
200	OK	Your message is valid, but it is not queued to be delivered.
202	ACCEPTED	Your message is both valid, and queued to be delivered.
400	BAD REQUEST	
401	UNAUTHORIZED	You do not have authorization to make the request.
403	FORBIDDEN	
404	NOT FOUND	The resource you tried to locate could not be found or does not exist.
405	METHOD NOT ALLOWED	
413	PAYLOAD TOO LARGE	The JSON payload you have included in your request is too large.
415	UNSUPPORTED MEDIA TYPE	
429	TOO MANY REQUESTS	The number of requests you have made exceeds the rate limitations.
500	SERVER UNAVAILABLE	An error occurred on the server.
503	SERVICE NOT AVAILABLE	The Email Gateway v3 Web API is not available.

CHANNEL PROVIDER STATUSES

+ Insert X Delete Export Refresh

Code
200
202
400
401
403
404
405
413
415
429
500
503
bounce
delivered
dropped

5 10 20

Personalized Content Management

The Hyper-personalization Automation processor empowers you with the ability to create effective and user-tailored ways of interacting with the customer. It can also be customized to better suit your needs if information should be extended to a desired communication channel.

FintechOS grants access to intuitive omni-channel content templates that can be personalized with dynamic tokens.

The Hyper-personalization Automation provides the following main content management features:

- Improved usability, with user interface complemented by tips & tricks and usage guidelines;
- Remaining characters counter based on defined message length;
- Tokens available for multiple data sources, available for insert directly in the context of the content template;
- Formatted tokens for numeric and date fields based on content culture;

- Approval workflow for managing content templates authoring;
- Preview messages exceeding the allowed length, during campaign simulation.

This section provides information on how to configure and use the personalized content features to their fullest extent:

- [Personalized Content Types](#) - allow you to have different types of content templates;
- [Content Settings](#) - allow you to automatically return attribute values in the content templates;
- [Personalized Contents](#) - allow you to start digital content campaigns with quick reusable content templates.

You might have communication channels to which you need to send more information to be used on the destination. You can do so by extending the app. For information on how to do this, see [Extend Personalized Content Management](#).

Content Settings

Content tokens (herein referred as content settings) are an easy way to further personalize content in business audience outreach. They can be used in the HTML body of a personalized content item in order to act as a variable that returns an attribute associated with the source entity. More specifically, this can be used to make a single personalized content that when read by the customer will display their name, instead of a generic greeting. This functionality can be used on many different kinds of attributes in order to create a truly impressive experience and display precisely the kind of information a customer needs.

Content settings can be easily managed and empower the business user to channel their creativity towards meaningful content creation.

Add Content Settings

To add a content setting, follow these steps:

1. Click the main menu icon at the top left corner.
2. In the main menu, click **Hyper-Personalization > Content Settings**. The **Content Tokens List** page opens.
3. Click the **Insert** button at the top right corner of the page. The **Add Content Token** page opens.
4. Fill in the fields.

Field	Description
Name	The name of the token that will be used in personalized content. This field is mandatory.
Context Entity	The name of the entity associated with the required data. This field is mandatory.
Attribute Name	The name of the returned data. This field is mandatory.
Description	Description of what the token is and what it might be used for.

ADD CONTENT TEMPLATE TYPE

CONTENT TEMPLATE TYPE

Name

5. Click the **Save and close** button at the top right corner to save the content token The record is added to the **Content Tokens List** page.

Delete button at the top right corner of the page. A confirmation dialog appears. Click **Yes** to delete the selected record.

Managing Personalized Content

Communicating with customers on a personal level is critical, but getting too personal can be very intrusive. Marketers need to be able to keenly interpret available customer data to extract the right insights.

Governance around the frequency of marketing messages and degree of personalization is also essential. In order to support such requirements, and help businesses overcome the challenges of the digital age, FintechOS provides a powerful tool to create and manage customer engagement campaigns, giving you the means to interact with customers in a meaningful manner.

Personalized content templates are essential to the content creation process. They allow you to remain on track by guiding you with useful fields, and minimizing the chance of mistakes by omission, and can also help the content team think strategically and holistically about the content they're creating for each page.

View Personalized Content Templates

To view the list of personalized contents, follow these steps:

1. Click the main menu icon at the top left corner.
2. In the main menu, click **Hyper-Personalization > Personalized Content**.

The **Content Template List** opens. If personalized content has already been created, this page displays relevant information: the ID and name of the template to quickly identify and distinguish between campaigns. The template type is also displayed along with the user who created it, the date, and the content template status.

This page provides you with the means to add, edit, or delete personalized contents.

Add Personalized Content Templates

NOTE

In order to create a personalized content template, you should first create content types. For information on how to do this, see [Creating Personalized Content Types](#).

To add personalized content, in the **Content Template List** page, click the **Insert** icon in the top right corner of the page. The **Add Content Template** page opens.

Adding a personalized content template is a two-step process.

1 Define content template

By default, the **Add Content Template** page opens on the **Define** step.

Fill in the fields:

Field	Description
Name	The name of the content template. This field is mandatory.
Reference Id	The number associated with the content template.
Template Type	Select between existing content types in order to categorize.
Attachment	Add an attachment to the content template.

ADD CONTENT TEMPLATE

+ Tips & Tricks

- define a template item for each **channel** and **language**
- organize your templates under the same **template type**

Name *

Reference Id

Template Type ↓

Created On

Created by user

Attachment or Drop file here

Click the **Save and reload** button at the top right corner of the page to save the record. The page reloads and the **Add Content Template** page opens.

Continue with the next step.

2 Manage content template items

Click **Manage Template Items** and follow these steps:

1. Select the **Communication Start Hour**, the hour when the communication can start.
2. Select the **Communication End Hour**, that is the hour when the communication ends.
3. Add content template items to differentiate between different user types and cultures in order to deliver personalized content.

Content templates contain multiple content template items. Each content template item represents the structure of the message to be sent, it contains fields like name and subject. These items also help determine the most important thing about communication campaigns, namely customer personalization done via the culture and channel fields, and can determine the specificity and period of messages.

Another important field is the body, where the template is rendered and can be modified. A very useful tool to have are the tokens, these are used to personalize the message with name, address, etc. The last checkboxes provide support for textual or HTML type of messages or enable analytics via tracking. Content templates are used to provide audiences with personalized content in a quick and efficient manner.

To add a content template item, follow these steps:

1. Click the **Insert** button. The **Add Content Template Item** page appears.
2. Fill-in the fields:

Setting	Description
Name	The name of the content template item. This field is mandatory.
Subject	The main content of the message. This field is mandatory.
Channel	<p>Select a communication channel (e.g., mail, SMS, etc.). This field is mandatory.</p> <div style="background-color: #e6f2ff; padding: 10px; border-radius: 5px;"> <p>NOTE</p> <p>You need to add channels (main menu > ADMIN > Omnichannel Communication Automation > Communication Channels) in order to be able to select one in this field.</p> </div>
Culture	Select the nationality of the user.
Max Message Length	The maximum length of the message to be sent. It is important for channel SMS.
Characters Remained	The number of remaining characters.

Setting	Description
Body	The main body that displays the message that will be sent. This field is of type HTML. Use the HTML Editor toolbar to format text as per your preference and insert tokens within the body by clicking Tokens from the editor's toolbar and selecting the desired token .
Send Message as Text	Switches between HTML and text.

ADD CONTENT TEMPLATE ITEM

Name Templateltem

Subject Discounted interest rates

Channel Email Culture English GB

Max. Message Length 200 Characters Remained -40

File - Edit - View - Insert - Format - Tools - Table -

Dear {Content_Template_Item.Name},

It is our pleasure to inform you that you've launched a new loan program.
The offer is valid by the end of the month.
Apply now for a loan and benefit of our limited offer.

Kind regards,
The Bank Team

Send Message as Text

3. Click the **Save and close** button at the top right corner to save the content template item.
4. Continue adding as many content template items as you need, then click the **Save and close** button at the top right corner to save the content template.

Extend Personalized Content

You might have communication channels to which you need to send more information to be used on the destination. You can extend the personalized content with the following types of information:

- **Statics** – The values are added on content item and are persisted with the same values when a campaign uses the content.
- **Dynamics** – On content item level, the user will add information as tokens and the values are composed from audience at runtime moment (when a campaign is launched).

In order to extend the information on a desired communication channel (add extended properties), you need to add in message a generic attribute formatted as key/value pairs.

To extend the information on a desired communication channel, follow these steps:

1. Create a new entity and add all the needed attributes. We recommend you to use the following naming convention: `FTOS_CMB_CommChannel_[channel name]`.

NOTE

The new entity should meet the following prerequisites:

- Contains at least one attribute lookup to `ebs.FTOS_CMB_ActionTemplateContent` named `contentItemId`.
- Static attributes are added as standard entity attributes.
- Dynamic attributes are added as lookup to `FTOS_CMB_ContentToken`.

2. On the desired communication channel, choose the newly created entity as value for `extendedPropertiesEntityId`.

3. On the content item, if the selected channel is the one that you set up, add all the desired values to be used by campaigns.

On the FTOS_DPA_MessageQueue entity, you will find for each message which uses the below channel, an attribute named ChannelProviderParams with all key/value pairs.

Extended Properties Example

Let's assume that you need to communicate from a campaign custom information to an endpoint on a mobile channel.

This section shows you how to extend Personalized Content to communicate from a campaign to an endpoint on a mobile channel the following custom information: first name, last name with some extra details like campaign type, availability date, and maximum credit amount:

1. Create an entity named FTOS_CMB_CommChannel_MobileApp with the following attributes:
 - FTOS_CMB_CommChannel_MobileAppid - PK
 - contentItemId – lookup to ebs.FTOS_CMB_ActionTemplateContent
 - campaignType – OptionSet
 - availabilityDate– Date
 - maxCreditAmount– lookup to FTOS_CMB_ContentToken
2. Add the Mobile App communication channel (**ADMIN > Omnichannel Communication Automation > Communication Channels**) and make sure that for the Extended Property attribute you select FTOS_CMB_CommChannel_MobileApp.
3. Create a new content and a new content item by following these steps:

- I. On the content template item, choose the **Mobile App communication** channel.
- II. Click the **Edit Channel Extended Properties** button. The **Edit AppMobile Extended Properties** page opens.
- III. From the **Content Item ID** field, select the campaign type (Notification).
- IV. From the **tokenDate** field, select a fixed date (e.g.,30.11.2019) and from the tokens list, select **MaximumCreditAmount**.

NOTE

You must add the token from the Audience.

- V. Save the form and return to the content template item.
4. Confirm the template and use it in a campaign.
5. Start the campaign.

For each attendee, you will find a message in the **Message Queue** entity. In **ChannelProviderParams** you will find a key/value pair; for example: {campaignType: notification, availabilityDate: 31.03.2019, maxCreditAmount: 2500}. You can use these values to send a proper message to the mobile app endpoint.

Task Management

Task management tools are designed to help you organize, prioritize, manage estimations, or visualize different projects. The FintechOS Task Management solution is beneficial for financial institutions as it provides the tools necessary to plan and track different banking and insurance workflows.

Task Management

FintechOS Task Management is designed to provide the tools necessary to organize allocation, prioritization, and oversight for workflows that operate with tasks. Such workflows are procedures undergone within banks and insurance companies, usually when they involve certain checks that must be done when onboarding new customers, whether when creating an account or contracting an insurance policy.

One such common procedure is compliance verification on a loan application. If during the loan application journey, the applicant is found within the bank's risk list, a series of verifications need to be done, sometimes by several individuals from the bank's compliance department.

Task Management accommodates this procedure by automatically defining queues based on several criteria and assigning such tasks to certain operators, based on their competence levels. This is all configured once within the journey and done automatically every time an application from a customer in the risk list is received in the system. However, this is only one use case. Task Management can be used in a number of scenarios depending on your company's business needs.

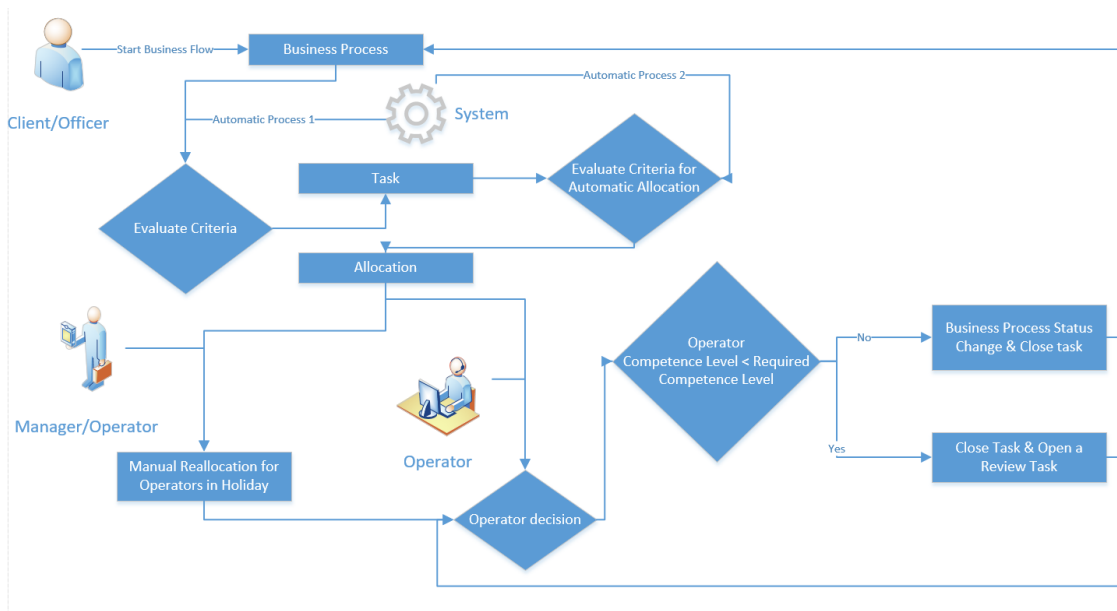
Task Management Features

Task Management has two major components:

1. **Task Management Dashboard** - this is the work environment for operators where they can view, review, and submit resolutions for the tasks.
2. **Task Management Admin Menu** - this is the menu where admins can configure operators, queues, profiles, competence levels, and filters to define the allocation process.

Task Management Flow

The following diagram is a high level understanding of what **Task Management** module addresses. The diagram showcases a number of automatic processes that occur when allocating a user to a specific queue.



In some digital processes, there is a need for action or decision from certain departments in order to fulfill a business need. In this context, the Task Management module can be configured to meet multiple such requirements.

Certain components need to be configured:

- Queue Type
- "Queues" on page 395
- "Operators" on page 401

- ["Competence Levels" on page 402](#)
- ["Filters" on page 403](#)
- ["Profiles" on page 404](#)

Creating Queues and Tasks

The **Queue Type** is designed to separate technical areas related to the data model. Several queues can be defined with the same queue type that can meet different criteria. In addition, filters can be defined to identify a specific queue.

For example, a loan origination journey would be split between personal loans and mortgages because different back-office departments handle such loans. To do this, we can create two queues, one with the filter specifying term loan as the product type, while the other with the filter mortgage as the product type. Both queues would use a queue type named retail loans with the master entity Retail Loan, and both would use the same form to display information to the back office user in an aggregated way. In this way, we make sure that mortgage applications are assigned to the right compliance officer, the one that would not usually handle personal loan applications.

FintechOS SDK identifies the queue based on the queue type and queue filter. Using the **RecordID** information and processor settings belonging to the Task Management, Innovation Studio can translate record values into queue filter and competence level files of type .json. Using this information, the proper queue is identified and the needed competence level is automatically set. This step is represented by the Automatic Process 1 in the above diagram.

After the queue is identified, the task is automatically created, and its value is stored in the **FTOS_CMB_QueueItem** entity. The task's status is set to **New**. Referring to the SDK, the task is created with the **FTOS_CMB_TaskManagement_CreateAndAllocateQueueItem** endpoint that identifies the right queue and creates a task with the required competence level.

Allocating Tasks to Operators

Another automatic process allocates the task to an available operator. This process needs the following configurations:

- Profiles with specific filters;
- Attached operators to queues.

The automatic allocation process considers the previously identified queue and performs the following iterations:

1. If the task is in the **Review** status, the process searches for an operator with the required competence level. If no operator is found, the allocation is not done, because tasks with the **Review** status must be processed only by operators with a specific competence level.
2. If the task is in the status **New**, the algorithm of allocation executes the following steps in the specified order. If an operator is found regardless of the step, the algorithm stops:
 - a. The system searches for old tasks with the same **RecordID**, orders them by creation date (**createdOn**) in a descending order, gets the operator from the first one, the first in the list, and checks if the operator is available.
 - b. If the **ReturnToSameOperator** flag is set to *true* for the queue, the algorithm searches for a task with the same Unique ID in the last X months (system parameter: **LastXMonths**) and if the operator is available, the allocation is made.
 - c. Identifies all available operators with the required competence level that are attached to the task's queue. From this list of operators, only those with the number of active tasks allocated less than max allocation active task (if any is set) are selected and they are ordered ascending based on the last allocation date (attribute for each **Queue** attached to **Operator**).
 - d. If an operator isn't found and the required competence level has a replacement competence level, the system tries to find an available operator using the **Replacement Competence Level**, and takes into consideration the maximum number of active tasks that an operator has (if any is set), and orders operators by last allocation date.

If an operator is found, the task's status is automatically changed to **Allocated** and the operator assigned to the task. For the operator, the fields **Last Allocation Date** and **Item Count** are updated. The task is visible on the **Task Management** dashboard tab **My active task** of the operator identified and allocated to the task.

If an available operator isn't found, then the task remains unallocated and can be seen in the **Task Management Dashboard** on the tab **Unallocated Tasks** by all the operators that are attached to that queue, and have the proper profile and proper competence level. From this tab, the operators can **Pick Up** tasks and the system changes the status of the picked up tasks to **Allocated**. An admin user or manager can also manually allocate an operator by accessing tasks through the menu **Queue Item**, open a task and manually selecting the operator. When the operator is saved on the task, the system automatically changes the status of the task to **Allocated**. The **Last Allocation Date** and **Item Count** for an operator is automatically updated in the **Queue** context.

If the **Holiday** flag for an operator is set to *true*, then all their active tasks are changed to the status **AllocatedButHoliday** and they appear in the **Task Management Dashboard** on the tab **My colleagues tasks**. These tasks are visible to the following:

- all operators that are designated as replacement for the operator on holiday;
- all operators that have the profile matched with the one defined as replacement.

If an operator decides to work on a task allocated to another colleague, who, for example, may be on holiday, they can **Pick Up** the task. The system closes the task picked up with the status **ClosedReallocated** and a new task is created and allocated to the operator who picked it up.

When an operator works on a task and gives a resolution, the business process changes the status to **Closed**. The system automatically determines if the operator who operated the task has a lower competence level than the required competence level and, if so, the task is closed and a new task is created with the status **Review**. The automatic allocation process tries to identify operators who are available, have the required competence level and have the number of active tasks less than max allocation active task (if any are set). They are ordered ascending by last allocation date and get the first in the list. Tasks in status **Review** can be seen on the **Task Management Dashboard** on the tab **Tasks that need review**.

An operator can be attached to multiple Queues and for this reason work priorities need to be set. For this purpose **Task Management Dashboard** grids use the **Priorities for Queue** attribute to order the tasks for operators and further on, for a particular **Queue**. Tasks are ordered by using the **Queue Priority** configured for the **Queue**. For an operator, all tasks from all queues are presented in the same grids and each task opens the designated entity with the configured edit form.

The steps for creating and configuring queues, operators, filters and so on are detailed further in this guide.

Installing Task Management

Task Management must be installed on Innovation Studio and a number of other configurations must be done before being able to use the package.

Prerequisites

- FintechOS Platform minimum version v21.1.2

Importing the User Role

In order to use the Automation Blocks module, a specific user role must exist on the Innovation Studio instance. The .xml files needed for the **Task Management Role** are available in the **SecurityRole Data Import Templates** folder on the package. Follow the steps detailed in the [Importing a Deployment Package](#) page.

Install Task Management

Before attempting to install the **Task Management** module, you must cleanup the FTOS_CMB_OperatorXQueue existing data so a unique constraint can be added for Operator id and Queue id.

The next steps are:

1. Run the following file `01.sample_install_MainXML_SQL_DataImportTemplates.bat`
2. Run `02.sample_install_xml_DataConfigDeploymentPackage.bat`.
3. In Innovation Studio, go to **Admin > Settings**, and check the **Use full width forms** box.
4. Go to **Admin > System parameters** and set a value to parameter LastXMonths.
5. Add a sequencer with code QITEMS for **Queue Items**.
6. In FintechOS Portal, define a flow settings for your digital journey if one was not defined.

- If a flow settings was not defined:
 - add a new flow settings and select the above digital journey;
 - add a new processor setting with the digital processor type TaskManagement;
 - add any settings as needed.
- If a flow settings file was defined:
 - open the existing one created for the above digital journey;
 - add a new processor setting with the digital processor type TaskManagement;
 - add any settings as needed.

Task Management Admin Menu

The Task Management Admin Menu is used for making the initial configurations, such as defining queues, creating users for operators, defining competence levels, and so on. This menu is designed mainly for administrators for initial configurations, but also for managing queues, competence levels, filters and more.

The Admin Menu is found in the FintechOS Portal:

1. In the FintechOS Portal, click the main menu. The list of items opens.
2. From the list of items, click Task Managemen The list of features opens.

To view instructions on how to configure each feature, check the table of contents below:

Queues

In the case of a loan origination journey, for user applications to be displayed as tasks and assigned to an operator, a queue must be created of a certain type. The first step is to create a queue type, then a queue with that type.

Creating a Queue Type

Queue types can be created for data retrieved from an entity, a data form, or for a certain digital journey linked to a specific banking product. For example, if you have multiple banking products defined, like a credit card and a mortgage loan, you can create queues for each of these individually and assign them to certain operators.

1. In FintechOS Portal, click the main menu and access **Task Management > Queue Type**. The list of defined queues types opens.
2. Click the **Insert** button to add a new queue type and fill in the following information:

Field	Description
Name	The name of the queue type.
Task Type	The type of task, pick between Compliance, UW, and Chaser.
Due Days	The number of days until the queue type must receive a resolution.
Main Entity	Select the main entity from which tasks are to be assigned to this queue type.
Master Form	Select the master form from the main entity.
Digital Journey	Select the digital journey for this queue type.
Display Info	Add a list of attributes to be displayed in .json format.

3. Click **Save** after adding the information.

The screenshot shows a configuration form with the following fields:

- Name:** DocsTest
- Task Type:** Compliance
- Main Entity:** FTOS_BNKAP_RetailApplicantData
- Master Form:** FTOS_BARET_RetailApplicantDataMaster
- Digital Journey:** MortgageUnassisted

Below these fields is a 'Display Info' section containing a table with one row:

1	
---	--

A blue 'Save' button is positioned at the bottom right of the form.

Creating a Queue

After the queue type was created, you need to create a queue and assign it to the type.

1. In FintechOS Portal, click the main menu and access **Task Management > Queue**. The list of defined queues opens.
2. Click the **Insert** button to add a new queue and fill in the information:

Option	Description
Name	The name of the queue.
Queue Type	The type of the current queue.
Return to Same Operator	Tick the box if you want tasks to return to the same operator after exiting another queue.

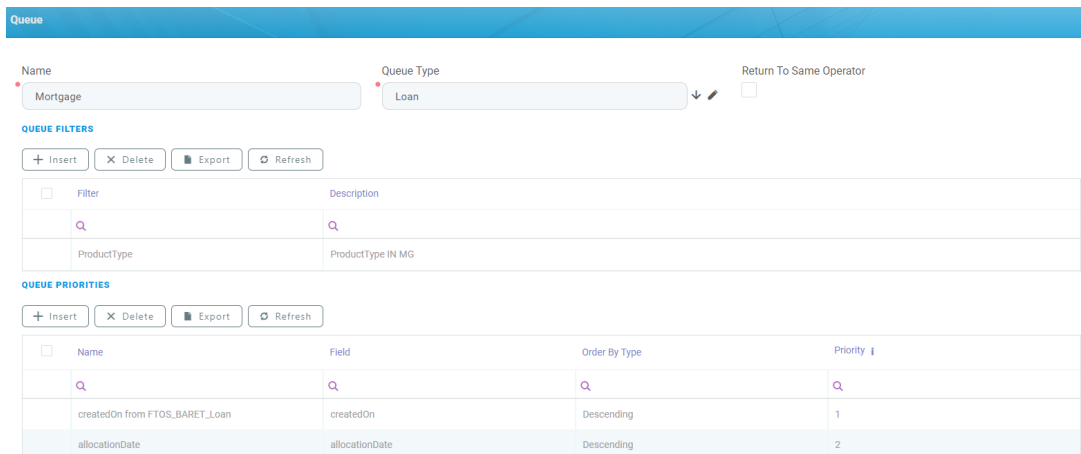
3. After adding the information, click **Save and reload**. The **Queue Filters** and **Queue Priorities** sections are now editable.
4. In the **Queue Filters** section, click **Insert**, pick one of the available filters, and add a description depending on the nature of the queue. Click **Save and reload**.
5. In the **Queue Priorities** section, click **Insert** and fill in the following information:

Option	Description
Name	The name of the queue priority.
Use Queue Main Entity	Tick the box to use the main entity of the queue.
Use Queue Item Entity	Tick the box to use the queue item entity.
Field	Pick a field based on which the priority is to be made.
Order By Type	Select an option between Ascending or Descending for ordering the data.

6. In the **OperatorXQueue** section, click **Insert** and fill in the following information:

Option	Description
Operator	Pick the operator from the list of all the available operators.
Competence Level	Set the competence level for the operator in this queue
Last Allocation Date	This remains blank.
Max Active Item Allocation Number	Can be used to limit the number of tasks an operator can receive from this queue.
Item Count	This remains blank.

7. Click **Save and close** to return to the **Queue** table.
8. The **Queue Items** section is automatically populated with items.
9. Click **Save and close**. The **Queue** window opens. Click **Save and close** to save your queue.



Managing Queue Items

The list of queue items is automatically populated with applications after Task Management is set up. However, there are cases when certain details about a specific queue item may need to be changed, such as the queue item to be renamed or allocated to a different operator. To manage queue items, follow these steps:

1. In FintechOS Portal, click the main menu and access **Task Management > Queue Item**. The list of available queue items opens.
2. Double-click a queue item and the following options open:

Option	Validation	Description
Name	Editable	The name of the queue item.
Required Competence Level	Editable	The needed competence level to review the task. This is calculated automatically by the rules set for the allocation process but the admin can edit the required competence level for specific items.
To Review	Read-only	Tick the box if the queue item needs review from the original competence level in case it was reviewed by a replacing competence level operator.

Option	Validation	Description
Returned	Read-only	Tick the box if the queue item was returned to the original operator, in case there were additional documents or information needed to finalize the request.
Operator	Editable	The current operator assigned to that task. The admin can use this field to manually set a different operator for that task.
Operator Profile	Editable	The operator's profile.
Operator Competence Level	Editable	The operator's competence level.
Task Date	Read-only	The date and time when the queue item was created.
Allocation Date	Read-only	The date and time when the task was allocated to the operator.
Resolution Date	Read-only	The date and time of the last resolution set on the task.
Replacement Operator	Read-only	This field is filled if a replacement operator has been assigned with that task.

Option	Validation	Description
Previous Queue Item	Editable	This field helps identify previous tasks that were correlated to the current one. For example, if a task was sent back to the originator for further clarification using the Request More Documents feature, when a new task is created, the previous queue item has the name of the related task.
Description	Editable	The description of the queue item.

NOTE
If the business status of the queue item is closed, then all fields are read-only.

3. Click **Save** after changing the information.

QUEUE QueueTest1 RECORDNAME Loan Application 0002545 RECORDDATE 13/04/2021 14:51

Queue Item

Name: Queue Item 0000418

Required Competence Level: Level3 on QueueTest1

To Review:

Returned:

Operator: [Redacted]

Operator Profile: Profile3 - CS Default & RiskList - BlackList

Operator Competence Level: Level 1 on Mortgage

Task Date: 13/04/2021 14:53

Allocation Date: 14/04/2021 10:22

Resolution Date: [Empty]

Replacement Operator: Select a value...

Previous Queue Item: Queue Item 0000417

Description: {"name":"Loan Application 0002545";"loanAmount":"640000";"financedAmount":"320000";"downPayment":"320000";"interestRate":"7"}

Operators

For tasks to be automatically assigned to a bank employee, operator profiles must be created in the system. These operator profiles hold information like the system user, whether they are on holiday or not, the last task allocation date, competence level, and so on. Follow the steps below to create an operator profile.

1. In FintechOS Portal, click the main menu and access **Task Management > Operator**. The list of defined operators is opened.
2. Click **Insert** to add a new operator and fill in the following information:

Option	Description
Name	The name of the operator.
Security User	The security user assigned to this operator.
In Holiday	Tick the box if the operator is in holiday.
Last Task Date	The last date when the operator was assigned a task.

3. In the **Operator Replacements** section, click the **Insert** button to add an operator that would replace the current one in case they are not available. Fill in the following information:

Option	Description
Name	The name of the replacement operator.
Operator	Pick a replacement operator from the available ones.
Queue	Pick the queue for the replacement operator.
Replacement Operator	The replacement operator.
Profile	Pick a profile for the replacement operator.
Valid From Date	The date from which this operator is a replacement.
Valid To Date	The date until which this operator is a replacement.

4. Click the **Save and close** button.

- In the **OperatorXQueue** section, click **Insert** to add one or more queues to the current operator. Fill in the following information:

Option	Description
Name	The name of the queue.
Queue	Pick the queue.
Operator	Pick a replacement operator from the available ones.
Competence Level	The competence level of the operator.
Last Allocation Date	The last allocation date.
Max Active Item Allocation Number	The maximum number of tasks that can be allocated to this operator.
Item Count	The number of tasks allocated to the operator.

- Click the **Save and close** button.
- In the **Profiles** section, click the **Insert existing** button to insert a profile for this operator.
- Click **Save and close**. The operator is created and is available in the list.

Competence Levels

Competence levels are used for defining the importance of tasks that are assigned to certain queues. Some tasks may need to be evaluated by operators with certain skill sets belonging to more than one queue. If a bank employee simply checks if the application has all relevant data, a compliance officer needs to verify if the applicant is in a risk list. For these cases, two competence levels may need to be defined.

- In FintechOS Portal, click the main menu and access **Task Management > Competence Level**. The list of defined competence levels is opened.
- Click **Insert** to add a new competence level, and fill in the following information:

Field	Description
Name	The name of the competence level.
Queue	The queue for this competence level.

Field	Description
Level	The level of this competence.
Replacement Competence Level	A replacement for the competence level.

3. Click **Save and close**.
4. In the **Competence Level Filters** section, click **Insert**.
5. Pick a filter from the list and add a description.
6. Click **Save and close**. The new competence level is now available in the list.

The screenshot shows the 'Competence Level' configuration page. At the top, there are three input fields: 'Name' (containing 'Level 1 on Mortgage'), 'Queue' (containing 'Mortgage'), and 'Level' (containing '1'). Below these is a 'Replacement Competence Level' field. Underneath is the 'COMPETENCE LEVEL FILTERS' section, which includes buttons for '+ Insert', 'X Delete', 'Export', and 'Refresh'. Below the buttons is a table with a search bar and a single filter entry: 'RequestedLoanAmount >= 0 - 300000'.

Filters

Some tasks can be assigned to different queues automatically, based on a certain filter such as the operator skill, competence level, the application date, the requested loan amount for a loan and so on. Filters are also used when defining competence levels.

1. In FintechOS Portal, click the main menu and access **Task Management > Filter**. The list of available filters opens.
2. Click **Insert** to add a new filter.
3. Fill in the **Name** of the filter and pick a **Filter Type**. Depending on the chosen type, certain options are displayed, such as **Entity View** or **Lookup** for a filter of type lookup.

The screenshot shows a 'Filter' configuration form with the following fields:

- Name:** Skill
- Filter Type:** LookUp
- Lookup:** FTOS_BNKAP_RiskList
- Entity View:** default

4. After completing the fields, click **Save and close**. The new filter is now available in the **Filter** list.

Profiles

Some operators have specialized skill sets and thus are required to review certain applications. For example, a certain operator verifies if an applicant is on the risk list, while another would approve the loan. Therefore, each operator must have at least one profile.

1. In FintechOS Portal, click the main menu and access **Task Management > Profiles**. The list of defined profiles is opened.
2. Click **Insert** to add a new profile and add a name.
3. Under **Profile Filters** section, click **Insert**.
4. Pick a filter for the profile and add a description.
5. Click **Save and close**.

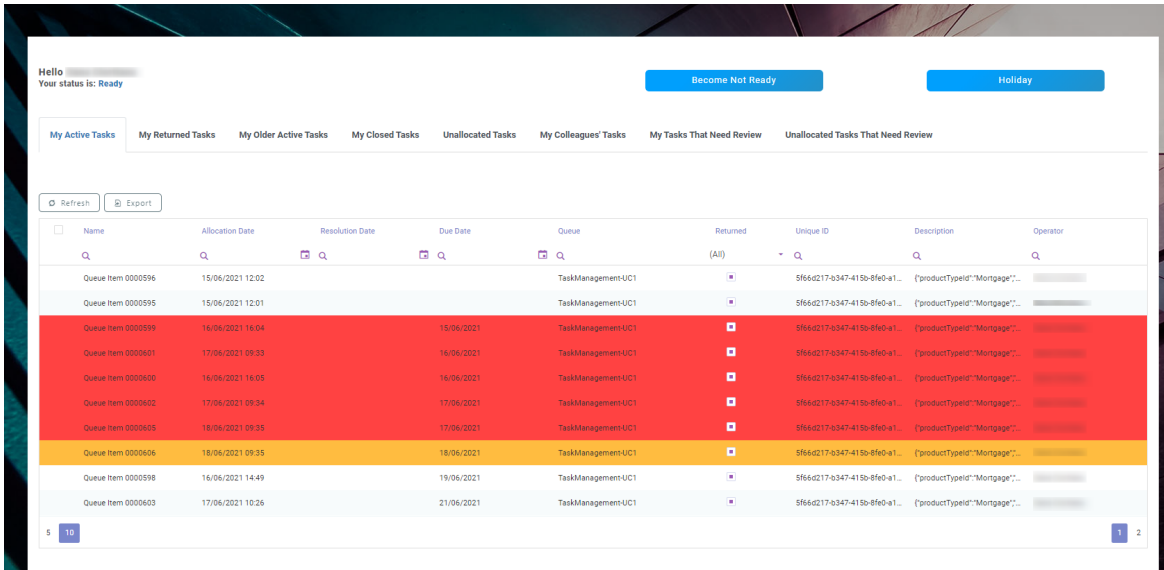
The screenshot shows a 'Profile' configuration form with the following elements:

- Name:** Demo profile 1
- PROFILE FILTERS:**
 - Buttons: + Insert, X Delete, Export, Refresh
 - Search bar: Description
 - Table of filters:

Description
Customer Segment IN Mass Market,Mass Affluent,Private Banking,New Customer
FTOS_BNKAP_RiskList IN Black List,High Risk

Task Management Dashboard

As opposed to the **Task Management Admin Menu**, the **Task Management Dashboard** is designed for operators to set their availability status, view their assigned tasks, or see previously processed tasks.



The **Task Management Dashboard** gives the operator the possibility to change their user's status by pressing one of the 3 buttons:

- **Become Ready:** changes operator status to **Ready** and all tasks with the **AllocatedButHoliday** status, associated with the operator, receive the **Allocated status**. In addition, if the operator's status is **Ready**, then new tasks are automatically assigned.
- **Become Not Ready:** changes operator status to **NotReady**.
- **Holiday:** changes operator status to **NotReadyHoliday** and all tasks with **Allocated** status, associated to the operator receive the **AllocatedButHoliday** status.

An operator can automatically receive tasks only if their status is **Ready**. The allocation process is done based on the operator. If an operator is available, they are allocated a task based on their competency levels. Still, it is advisable that when an operator opens the Portal, they should make sure to click the **Become Ready** button, and the **Become Not Ready** button when they leave work.

In the top right corner the user name and status is displayed, this can help confirm that the **Become Ready** button has registered properly and the operator is ready to receive tasks.

Dashboard Tabs

When a task is registered on the system it is automatically assigned to a queue and an operator based on some factors, such as: queue type, the operator profile, competence, and availability. For example, if during a loan origination journey, the applicant is found in a risk list, then their application is automatically assigned to a queue and an operator in the form of a task.

The Task Management dashboard contains 6 tabs:

- **My Active Tasks** tab: displays the tasks assigned to the current operator. It is the main tab for operators from where they can pick-up tasks that are allocated to them. The tasks found here can be either from the automatic distribution or manually assigned from admin.
- **My Returned Tasks** tab: displays tasks assigned to the operator and returned for various reasons.
- **My Older Active Tasks** tab: displays older active tasks that can be filtered by date of allocation or resolution.
- **My Closed Tasks** tab: displays closed tasks for the current operator.
- **Unallocated Tasks** tab: displays the list of unallocated tasks.
- **My Colleagues Tasks** tab: displays tasks allocated to colleagues who are on holiday. Use the **Pick-up** button to allocate a certain task to the current operator and change the queue item status from **Allocated** to **Closed** or **Reallocated**.
- **Tasks That Need Review** tab: displays in review allocated tasks to the current operator. They are ordered by queue allocation and based on the set queue priority.

Each tab has the option to export data from the tab or from all other tabs accessible to the current operator. When clicking the **Export** button, only the selected queue items are exported. The data is exported in .xlsx file types. The **Refresh** button can be used to manually refresh a list under a tab, but the refresh function is automatic by default and does this every 5 seconds.

Queue Item Details

The dashboard table contains a number of columns referring to details about the queue item, allocation date, operator, and so on. These columns are explained below.

Column	Description
Name	The name of the queue item.
Allocation Date	The date and time when the queue item was allocated.
Resolution Date	The date and time when the queue item received a resolution.
Due Date	The due date when the queue item must receive a resolution.
Queue	The queue to which the queue item belongs.
Returned	Shows whether the queue item was returned or not.
Unique ID	The unique id of the queue item.
Description	The description of the queue item.
Operator	The operator allocated for this queue item.

If a certain queue item is close to or surpassed its deadline, the due date for receiving a resolution, it is colored inside the table, to make it easier to spot and bring a certain urgency to the process:

- red: the queue item has surpassed its due date and needs immediate attention from the operator;
- orange: the due date is today and the queue item needs attention from the operator.

In order to change the colors, you need to change the rgba from the stylesheet `FTOS_CMB_TaskManagementDashboardStylesheet` on the following classes:

```
.taskManagementWidget .dx-row.dx-data-row.dx-column-lines.DeadlineNow{
  background: #ffa500bf;
}
.taskManagementWidget .dx-row.dx-data-row.dx-column-lines.DeadlinePassed{
  background: #ff0000bd;
}
```

Processing a Task

In the **Task Management Dashboard**, click any task from under a tab to open the following view:

AUTOMATION BLOCKS USER GUIDE

Application Documents

Name	Signed Document	Description
No data		

Task History

Queue Type	Allocation Date	Operator	Queue	Resolution Date
Compliance_Loan_RetailLoanApplication_BO	13/04/2021 16:45		QueueTest1	
Compliance_Loan_RetailLoanApplication_BO	13/04/2021 17:00		QueueTest1	
Compliance_Loan_RetailLoanApplication_BO	13/04/2021 18:04		QueueTest1	
Compliance_Loan_RetailLoanApplication_BO	14/04/2021 10:51		QueueTest1	

This view should be in connection to a digital journey so that details about applications can be displayed. For example, the previous view was configured in the context of the [Mortgage Loan Origination](#) digital journey, and it displays all information about the customer's application, such as applicant financial and personal data, the requested amount and results of calculations, property details, and so on.

Therefore, the operator can fully review the application and give their resolution in the **Compliance** tab:

- **Approve** - is a final resolution that approves the request and no further actions are required.
- **Reject** - is a final resolution that rejects the request. For this resolution there is a mandatory **Reject Reason** field that needs to be filled in by the operator.
- **Additional Documents** - is an intermediary resolution that pushes back the task to the user that originated it and requires additional information or documents for the operator to give a final resolution. After the information or documents are provided, the task returns to the same operator and can be found in the **My Returned Tasks** tab of the **Task Management Dashboard**.