Core Claims Admin 2.3.0

User Guide

TOC

Overview

Core Claims Admin is a solution aimed at operators who are directly involved in claims processing. The present guide is for setting up a **Core Claims Admin** solution on your **FintechOS Portal** environment.

An insurance claim is a claim for compensation over a loss. Usually, the claim is initiated by the insured party. The insurer checks if the loss event falls under the policy scope and validates the claim and, once approved, issues payment to the insured, or to an approved interested party on behalf of the insured. **FintechOS** clients use the **Core Claims Admin** solution to automate routines of this typical scenario, in order to increase the efficiency and accuracy of their claim processing operations.

The **Core Claims Admin** is comprised of a series of automated business steps that users follow through, from registering loss notifications, documenting claims, validating claims to, if approved, scheduling payments, in a timely and accurate manner. It also allows the insurer to maintain a constant communication with the insured.

When necessary, the **Core Claims Admin** solution can be personalized in **Innovation Studio**, according to specific requirements from insurance companies. Thus, implementation time is shortened, while you can make sure that each component fulfills your specific needs. Additionally, using **Core Claims Admin** along with different **FintechOS** Automation Processors enables insurance companies to also digitize other workflows and improve accuracy, while reducing the amount of time spent on routine business operations.

Business Pain Points

FintechOS clients use the **Core Claims Admin** solution to respond to different challenges related to:

- time-consuming routines;
- pressure to assess and validate claims within certain time limits;
- routines more prone to human error when done manually;
- a growing trend of claims falling outside the policy scope.

The **Core Claims Admin** solution makes places for better business performance when dealing with these pain points by enabling insurance companies to seamlessly automate their claim processing routines.

Core Claims Admin Key Features

The solution has the following key features:

- registration of loss notifications;
- generating claim statements;
- automated and, if necessary, manual validation;
- scheduling payments.

Core Claims Admin Key Benefits

The benefits of using the **Core Claims Admin** solution are the following:

- works easy with volumes;
- scales from simple claim models to complex;
- speeds up the validation processes by automating routines;
- reduces the risk of human error while handling accounts;
- frees time for where attention is really needed: prioritizing urgent claims.

HINT Integrate **Core Claims Admin** with more **FintechOS** solutions for insurance in order to make the best of process automation for your company, portfolios, products and clients!

Installing Core Claims Admin 2.3.0

Follow the guidelines below to install and configure **Core Claims Admin 2.3.0**.

Prerequisites

Before installing **Core Claims Admin 2.3.0**, make sure to install the following:

- **HPFI v22.1.0** or higher.
- **SySDigitalSolutionPackages v21.2.2301.zip** or higher.

Please follow the steps below to install **Core Claims Admin 2.3.0**.

Core Claims Admin 2.3.0 Install

Install the following packages in this exact order:

1. **Core Claims Admin2.3.0**
2. **Core Claims Admin 2.2.0 Import**.

Follow the installation steps below:

1. Install **Core Policy Admin 3.3.0**.
2. In **Innovation Studio**, import the **Core Claims Admin2.3.0** digital asset by following the steps described here.

3. After the import is complete, install **Core Claims Admin2.3.0** as described in the standard procedure.

> **NOTE**
> If there are any errors, the system displays a message with the reason why the package could not be imported. To see the warnings, consult the log of the selected deployment package. For more information, see Viewing Deployment Package Logs.

4. Add/modify the Vault keys:

- **app-settings Portal**:

  Identify the following keys and add their values with your SMTP information:

  {

  "baseUrlApi": "PORTALAPI_URL *",

  "clientApi": "yourClient",

  "userApi": "yourUserName",

  "passwordApi": "youUserPass",

  "SMTP:Port": "***",

  "SMTP:Host": "***",

  "SMTP:EnableSSL": "0",

  "SMTP:User": "***",

  "SMTP:Password": "***",

  "DefaultFromEmail": "***"

  }

  * URL of the portal site using EBSDefaultAuthentication = EBS

  *** = your SMTP information

- **app-settings Studio**:

Identify the following keys and add their values:

{

"SMTP:Port": "***",

"SMTP:Host":"***",

"SMTP:EnableSSL":"0",

"SMTP:User":"***",

"SMTP:Password":"***",

"DefaultFromEmail":"***"

}

*** = your SMTP information

- **app-settings Job server**:

{

"UploadFolder": "yourPath:\Sites\UploadEBS",

"AttachmentPath": "yourPath:\Sites\UploadEBS",

"FileUploadWhiteList": ".pdf,.doc,.docx,.els,.jpg,.jpeg,.xlsx,.dll,.ppt,.pptx,.txt,.png,.ttf,.xml",

"baseUrlApi": "PORTALAPI_URL *",

"clientApi": "yourClient",

"userApi": "yourUserName",

"passwordApi": "youUserPass",

"SMTP:Port": "***",

"SMTP:Host": "***",

"SMTP:EnableSSL": "0",

"SMTP:User": "***",

"SMTP:Password": "***",

"DefaultFromEmail": "**",

}

* URL of the portal site using EBSDefaultAuthentication = EBS

*** = your SMTP information

- **UploadFolder** , **AttachmentPath** - for a job server installed as an web app, UploadFolder and AttachmentPath keys are not needed. Instead, use the standard configuration steps to allow the job server access to the blob storage used by the other sites (to the same UploadEBS folder).

5. Follow steps 2 and 3 to import and install **Core Claims Admin Import**.
6. Approve the status of Insurance Parameters: Log in to **FintechOS Portal**.
7. Navigate to **Main Menu > Settings > Insurance Parameters**.
8. Select a parameter and open it for edit.
9. From top left corner change the parameter status from **Draft** to **Approved**.
10. Repeat steps from 12 to 14 for all Insurance Parameters in **Draft** status.
11. Done!

Security Roles for Core Claims Admin

FintechOS security architecture is a unified security design aimed at empowering **FintechOS** clients to address the necessities and potential risks involved in a certain scenario or environment. The security roles are an inbuilt part of the **Core DPA Platform** security architecture, designed to help you mitigate cyber crime related risks and keep data secure across all your business flows. Consequently, you use security roles to protect sensitive data and configure various organization layers to allow for better communication, collaboration, or reporting.

> **NOTE**
> For more details, see also the Default Security Roles documentation.

The following roles are available for Core Claims Admin in order for the users to be allowed to only perform the actions which are attributed to them:

Security Role	Description
FNOL User	This user only has the rights to insert in the FNOL list + Claim insert, at user level.
Claim superUser	This user only has the rights to insert new Claims and process the entire flow without the possibility to validate the loss values, approve claim payments and approve declining proposals.
Claim manager	This user has the rights to validate loss values, rights to approve claim payments and the rights to approve declining proposals.

The following are the defined security privileges per every role (where V=view, I=insert, E=edit):

Functionality	FNOL user	Claim superUser	Claim manager
FNOL	View Insert Edit	View	View

Functionality	FNOL user	Claim superUser	Claim manager
Claims	Insert	View Edit	View
Loss value valdiation	-	View Insert	View Edit (approval)
Payments	-	View Insert	View Edit (approval)
Declining proposals	-	View Insert	View Edit (approval)
Claim payments	-	View	View Edit (approval)
Reserves	-	View	View

The table below presents which menu items are accessible for every security role.

Manu item	FNOL user	Claims superUser	Claim manager
FNOL	x		
Claims		x	x
Reserves		x	x
Claims payments		x	x
Declining proposals		x	x

HINT

Apart from the **Core Claims Admin Security Roles**, you can always define new roles to meet your business needs. For more details, consult the Creating or Editing Security Roles documentation.

Solution Walkthrough

The **Core Claims Admin** solution is designed to offer you a streamlined route for claims processing. The solution is comprised of the following key flows:

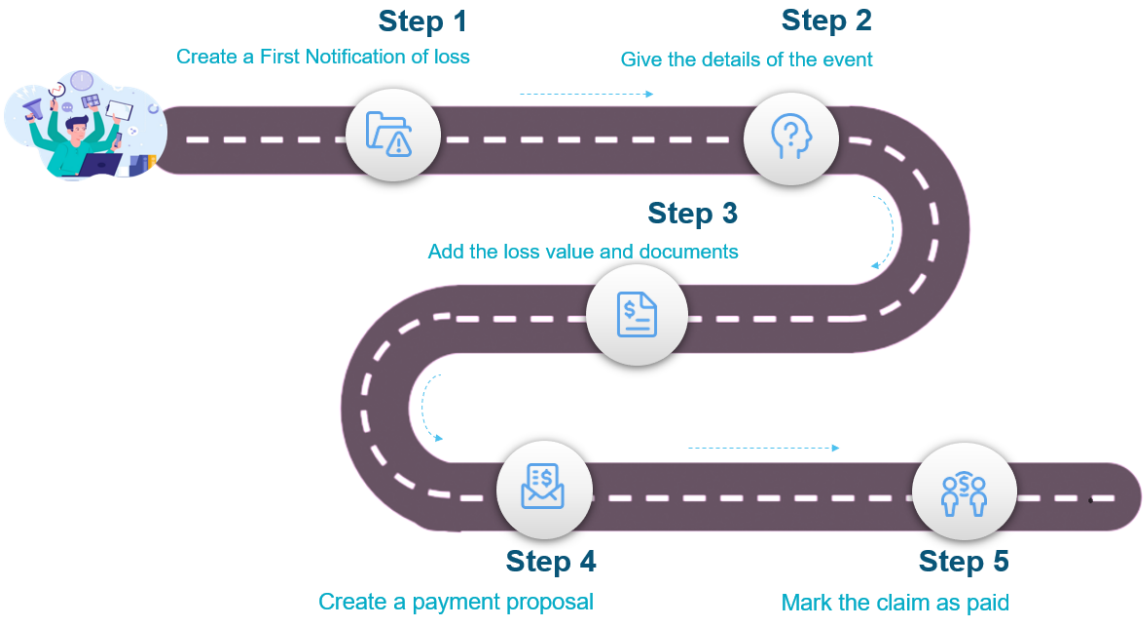
The First Notification of Loss flow - for collecting information about the policy, the notifier, the people involved, with the option to continue to **Create claim** or **Cancel notification** steps. Creating a **First Notification of Loss** is mandatory in order for any claim to be processed by the system. All the information that you insert in this step is re-used during the **Core Claims Admin** solution, in automatic auto-complete and read-only fields.

The Claim flow - for creating, documenting and managing claims, with upload options and tools to facilitate the assessment of the claim. A claim can be configured anytime after the FNOL is initiated. This **Core Claims Admin** configuration menu is comprised of eight tabs to help you complete the processing of the claim.

The Payment Proposal flow - for creating and scheduling payments. A payment can be configured anytime after the FNOL is initiated. The Payment flow has four steps and helps you manage any payment.

Rejection, Journal & Third Party Details - pages for adding supplementary information and managing the related documents and third party details, if applicable.

Below is an illustration of the **Core Claims Admin** process:



Core Claims Admin Key Steps

To process a new claim with the **Core Claims Admin** solution take the following steps:

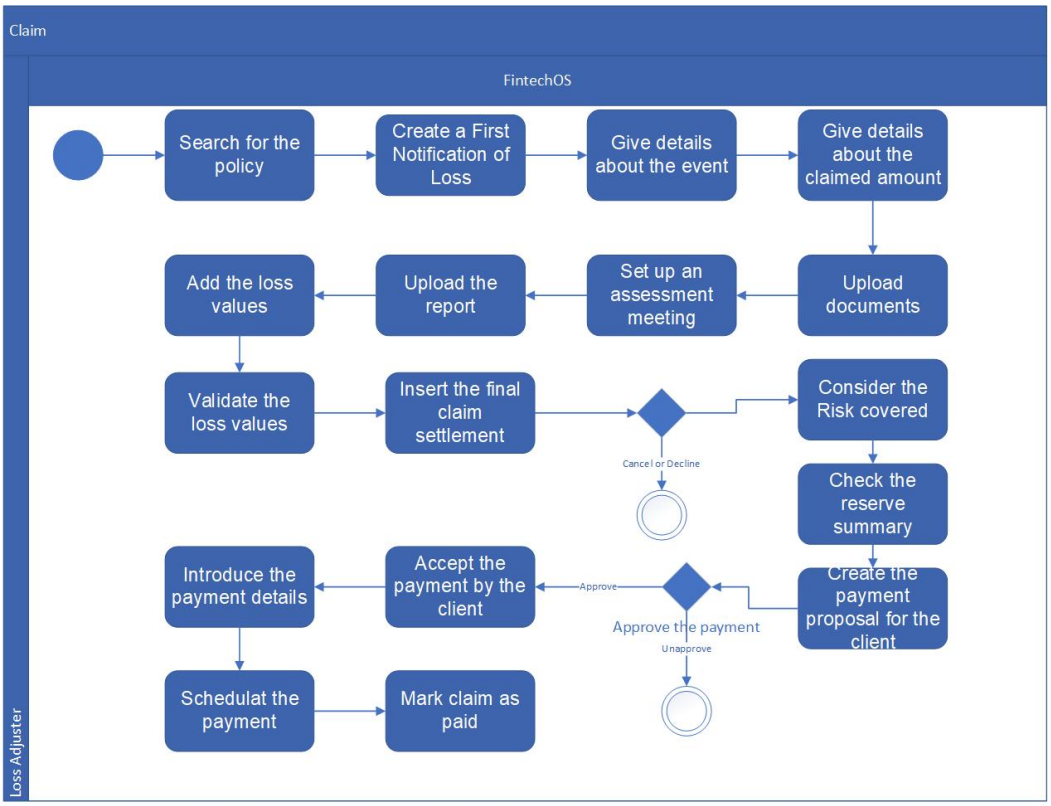
1. Open the **FintechOS Portal** and log in with your credentials.
2. In the menu, go to the **Claim > First Notification of Loss** list and click **Insert**, at the top right of the screen.
3. Create a **First Notification of Loss** by filling in the requested information and documents.
4. Use the **Claims** page to create the new claim record.
5. Use the **Assessment** page to upload the assessment documents and trigger validation. If validated, create a **Reserve**.

6. Create a **Payment Proposal** to send to the client. Register the customer's agreement, request **Payment Approval** and **Schedule** the transaction. Mark the claim as paid.
7. Insert information about the **Rejection** of the claim, manage related **Documents** and **Third Party** details, if applicable.

Also, check this video for a rapid view over the entire **Core Claims Admin** solution, for processing a valid claim.

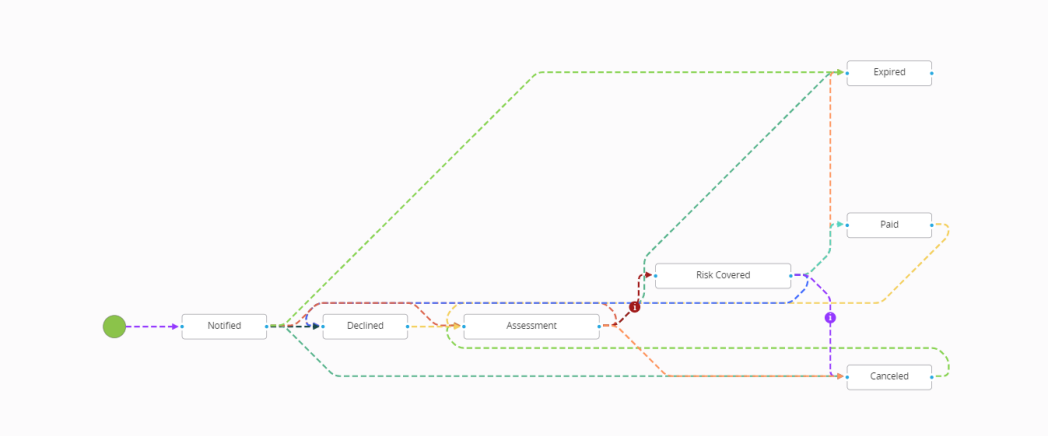
Business steps model

Below are the steps for the **Core Claims Admin** solution:



Business statuses model

Below are the statuses for the **Core Claims Admin** solution:



HINT Go to the First Notification of Loss page now to find out more about launching a new digital journey.

Create a First Notification of Loss

The **Core Claims Admin** solution for registering a new claim record starts from the **First Notification of Loss (FNOL)** page. You must log into the **FintechOS Portal**, go to the main menu on the left side of the screen, select the **Core Claims Admin** solution and double-click to open it. After launching the new solution, use the **FNOL** menu to insert information about the policy, the notifier, the people involved, and upload the requested documents.

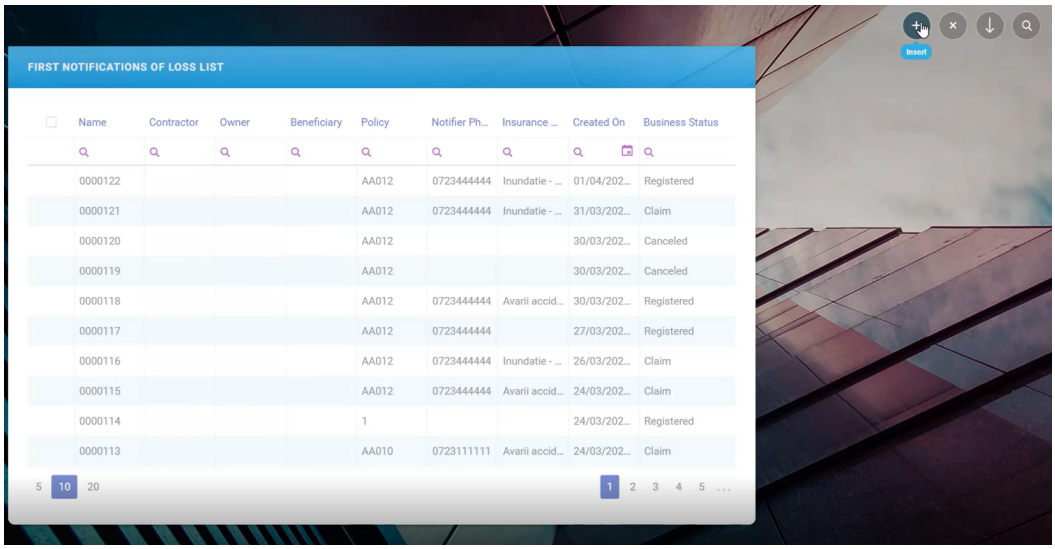
> **IMPORTANT!** A **FNOL** must be registered in order for any claim to be processed by the system.

Register a new FNOL record using the following steps:

FNOL general data

Inside the **Core Claims Admin** window, select **First Notification of Loss**.

The **FNOL List** page is displayed. Click **Insert** to add the a new **FNOL** for a policy.

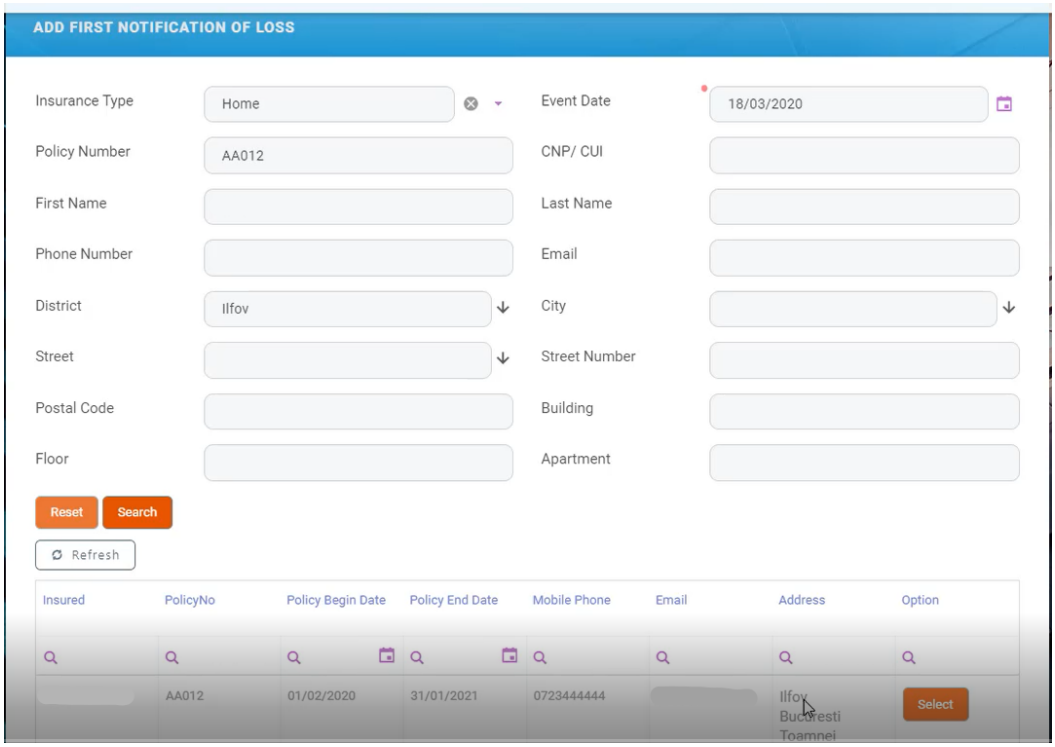


HINT All the information that you provide on this step is used to search and identify the policy record that you want to process. Pay attention to this step as the information you provide here will also be used in the next tabs, in automatic auto-complete and read-only fields.

Fill in the following general information:

Field	Description
Name	It is the first notification of loss number.
Contractor	It is the notifier's name.
Owner	It is the owner of the policy.
Beneficiary	It is the person who benefits of the money.
Policy	It is the policy number.
Notifier Phone	It is the phone number of the person to call.
Insurance risk	It is the risk from the policy.
Created on	The date when the FNOL was made.
Business Status	It is the status the FNOL has.

In addition, complete the fields with the details about the policy.



Filed	Type	Description
Insurance Type	Option set	It is the type of insurance for which the FNOL is opened.
Policy Number	Text	It is the policy number.
First Name	Text	it is the first name of the owner.
Phone Number	Text	It is the contact number.
District	Text	It is the district where the asset is located.

Filed	Type	Description
Street	Text	It is the street where the asset is located.
Postal Code	Text	It is the postal code where the asset is located.
Floor	Text	It is the floor where the asset is located.
Event Date	Date	The Event Date field is mandatory.
CNP/ CUI	Text	It is a unique number for fiscal enrollment (CUI).
Last Name	Text	It is the last name of the owner.
Email	Text	It is the email of the owner.
City	Text	It is the city where the asset is located.
Street Number	Text	It is the street number where the asset is located.
Building	Text	It is the building where the asset is located.
Apartment	Text	It is the apartment where the asset is located.

Click **Search** to display the existing policies based on the parameters from fields that you previously filled in.

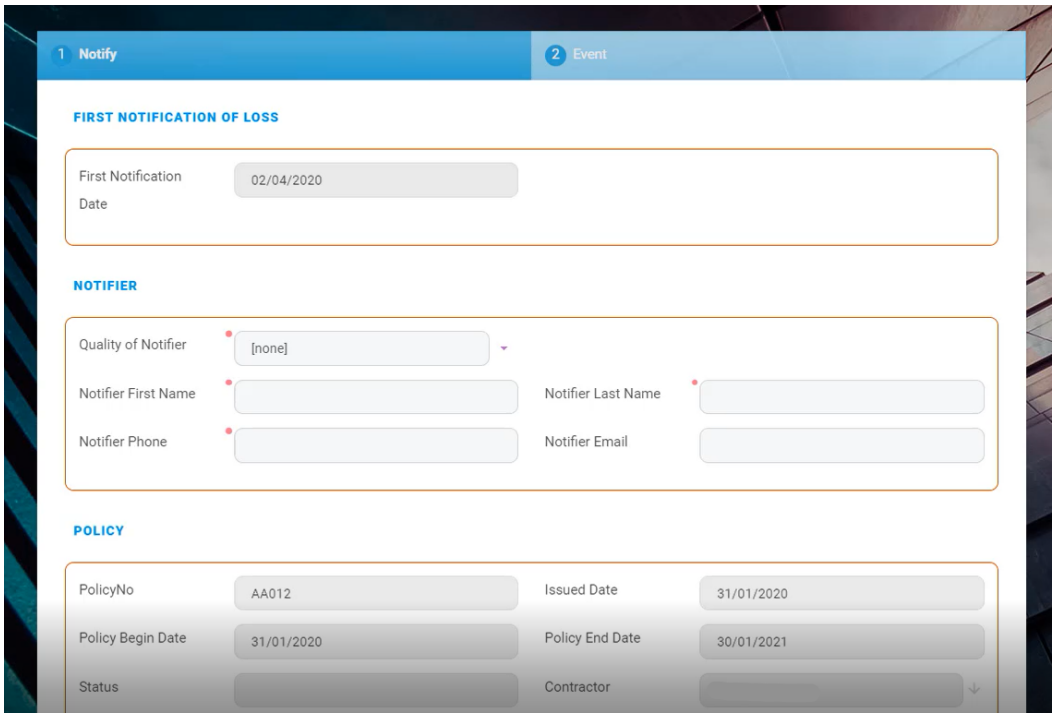
FNOL policy data

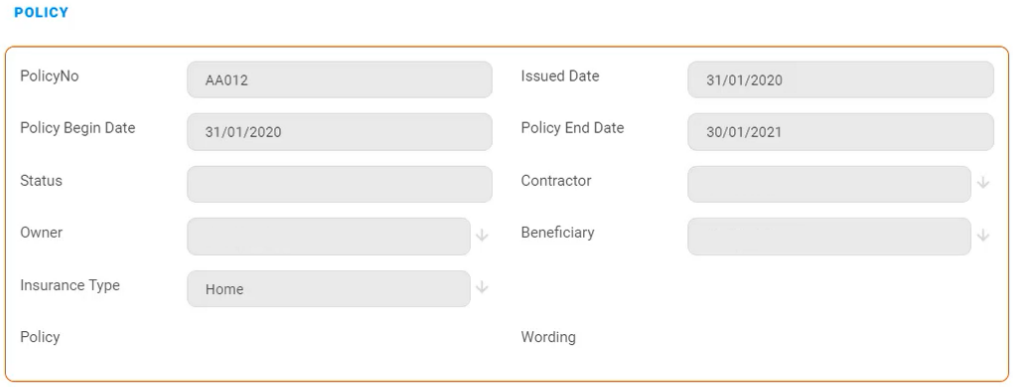
After clicking **Search**, you see a list of policies matching the data you introduced in the previous step. From this list choose the policy that you want to process and click **Select**. The next two tabs of the **FNOL** become available - **Notify** and **Event**.

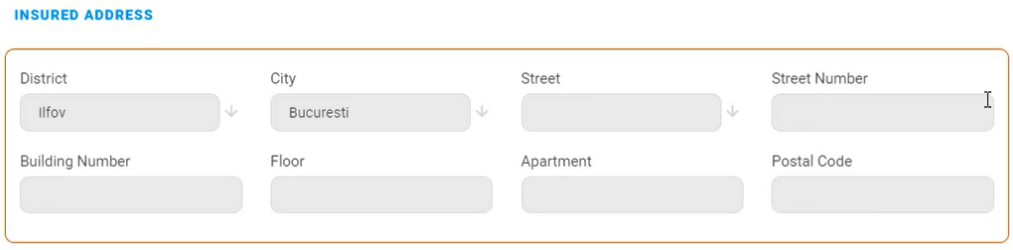
Go to the **Notify** tab and click on it. The **Notify** window is displayed. Here you find the following sections: **First Notification of Loss**, **Notifier**, **Policy** and **Insured Address**.

Check if there are any empty fields and complete those.

> **HINT**
> In the **Notify** tab, if you select the **Insured** option in the **Quality of Notifier** field, all the other **Notifier** fields become populated with data extracted from the policy. Also, the fields in the **Policy** section become automatically populated with the information extracted from the policy.







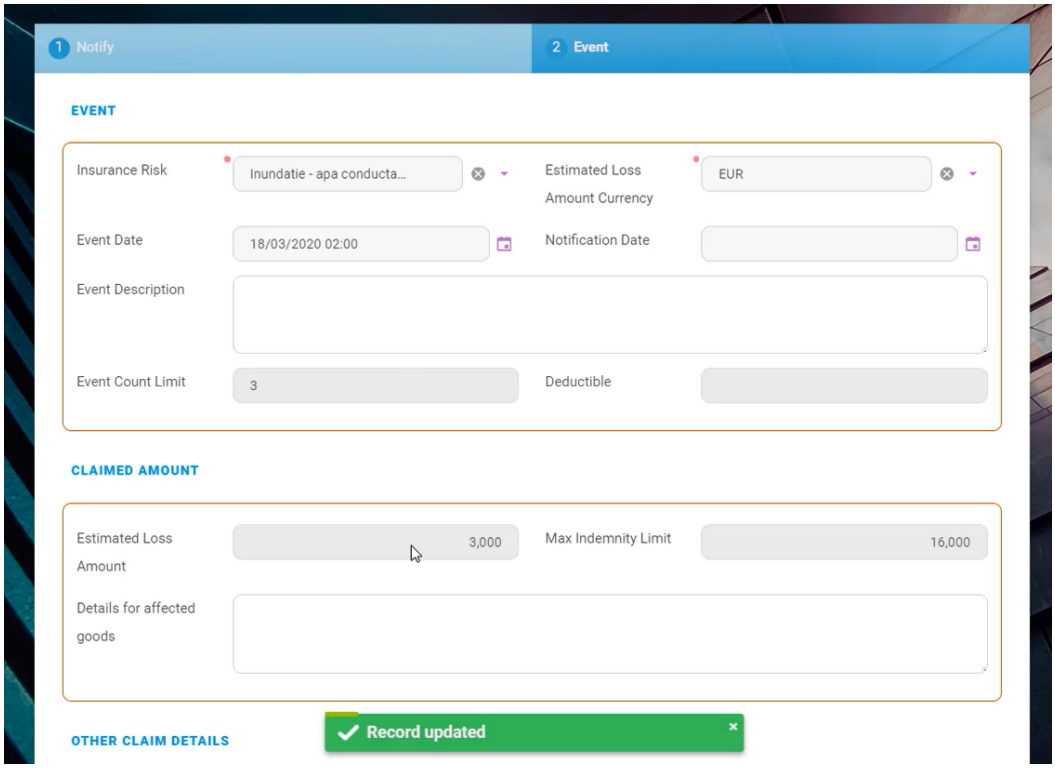
Here is a description of the four tables in the **Notify** tab:

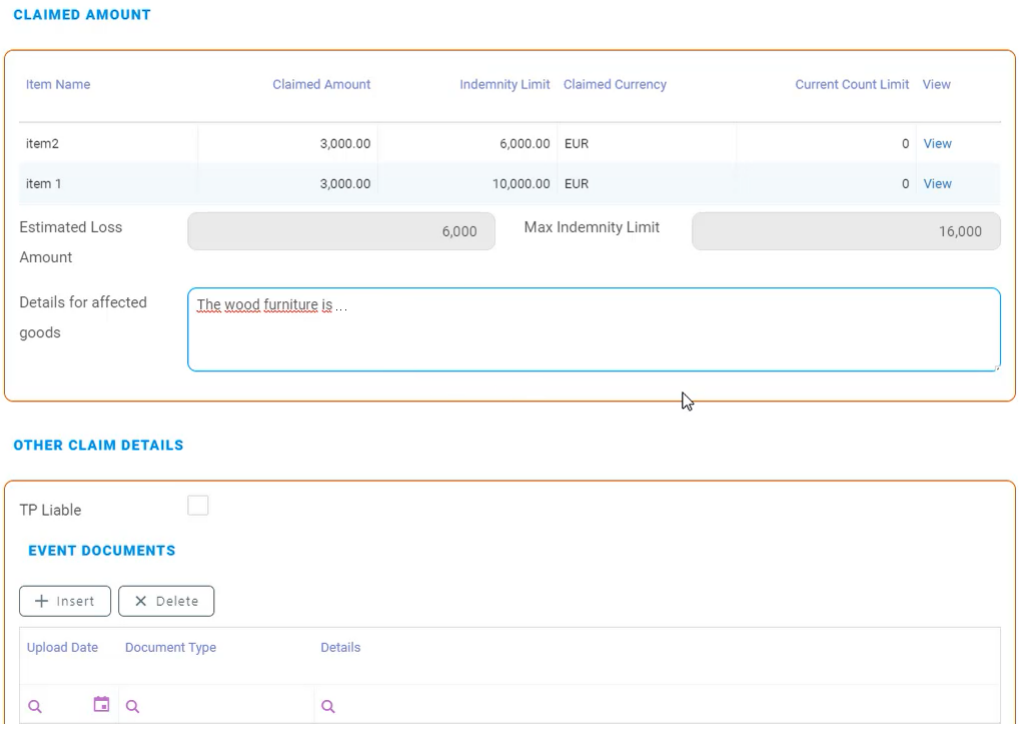
Field	Type	Description
FIRST NOTIFICATION OF LOSS		
First Notification Id	Text	It is read-only.
First Notification Date	Date	It is read-only.
NOTIFIER		
Quality of Notifier	Option set	It is the role of the person who notified the FNOF.
Notifier First Name	Text	It is the name of the notifier.
Notifier Phone	Text	It is the notifier's phone number.
Request Number	Text	
Notifier Last Name	Text	It is the notifier's name.
Notifier Email	Text	It is the notifier'snumber.
POLICY		
PolicyNo	Text	It is read-only.
Policy Begin Date	Text	It is read-only.
Status	Text	
Owner	Text	It is read-only.
Insurance Type	Text	It is read-only.
Policy		
Issued Date	Text	It is read-only.
Policy End Date	Text	It is read-only.
Contractor	Text	It is read-only.
Beneficiary	Text	It is read-only.
Wording		
INSURED ADDRESS		
District	Text	It is read-only.
City	Text	It is read-only.
Street	Text	It is read-only.
Street Number	Text	It is read-only.
Building Number	Text	It is read-only.
Floor	Text	It is read-only.
Apartment	Text	It is read-only.
Postal Code	Text	It is read-only.

Click **Save and reload** and proceed to the next tab.

FNOL event data

Click the **Event** tab. The **Event** window is displayed with the following sections: **Event**, **Claimed Amount** and **Other Claim Details**.





Check if there are any empty fields and complete those.

> **IMPORTANT!** In the **Event Description**, **Details for affected goods** and **Additional Comments** fields give as many details as possible. Completing those activates the next fields and you move forward through the journey.

Below is a description of the three tables in the **Event** tab:

Field	Type	Description
Event		
Insurance Risk	Option Set	Depending on the type of policy, the risks for each individual policy is shown.
Event Date	Date Time	It is the date of the event.
Event Description	Text Area	Here the operator can describe the event.
Event Count Limit	Whole number	It is the maximum number of risks the policy has. It is read-only.
Estimated Loss Amount Currency	Option Set	It is the currency.
Notification Date	Date	The date when the notification was made.
Deductible	Number	The deductible is the amount the client pays out of pocket in case of damage.
Claimed Amount		

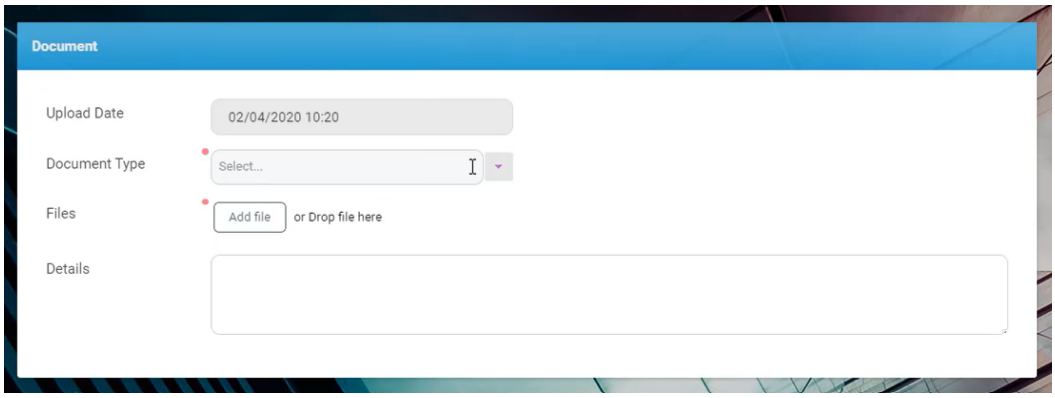
Field	Type	Description
Claimed Amount	Number	It is the amount of money the beneficiary wants.
Estimated Loss Amount	Number	It is the maximum amount of money the beneficiary can be given.
Estimated Loss	Number	It is the total amount of money the beneficiary wants.
Max Indemnity Limit	Number	It is the maximum amount of money to be given.
Details for affected goods	Text area	It is the description of the state of the goods.
Other Claim Details		
TP Liable	Bool	Third-party liability.
TP Liable Known	Bool	Check the box if the Third-party liability is known by the notifier.
Event Documents		You can attach documents sent by the notifier. It opens a new table.
Additional Comments	Text Area	For any details regarding the event.

In the **Claimed Amount** table, click **View** to open the pop-up table for the **Claimed Risk**.

Here is a description of the **Claimed Risk** view:

Field	Type	Description
Claim	Option Set	It is read only.
Policy Insured Risk	Option Set	It is read only.
Claimed Amount	Number	It is read only.
Claimed Currency	Option Set	It is read only.
Current Count Limit	Number	It is read only.
Current Value Limit	Number	It is read only.
Loss Amount	Number	It is read only.
Claim Currency	Option set	It is read only.
Indemnity Amount	Number	It is read only.
Indemnity Currency	Number	It is read only.
Indemnity Limit	Number	It is read only.
Risk Currency	Option Set	It is read only.

In the **Other Claim Details** table, click to upload the necessary documents.



After upload, two buttons become active: **Create Claim** and **Cancel Notification**.

HINT The next step of the **Core Claims Admin** solution is to create a **Claim** based on the registered **FNOL**. Press Create Claim.

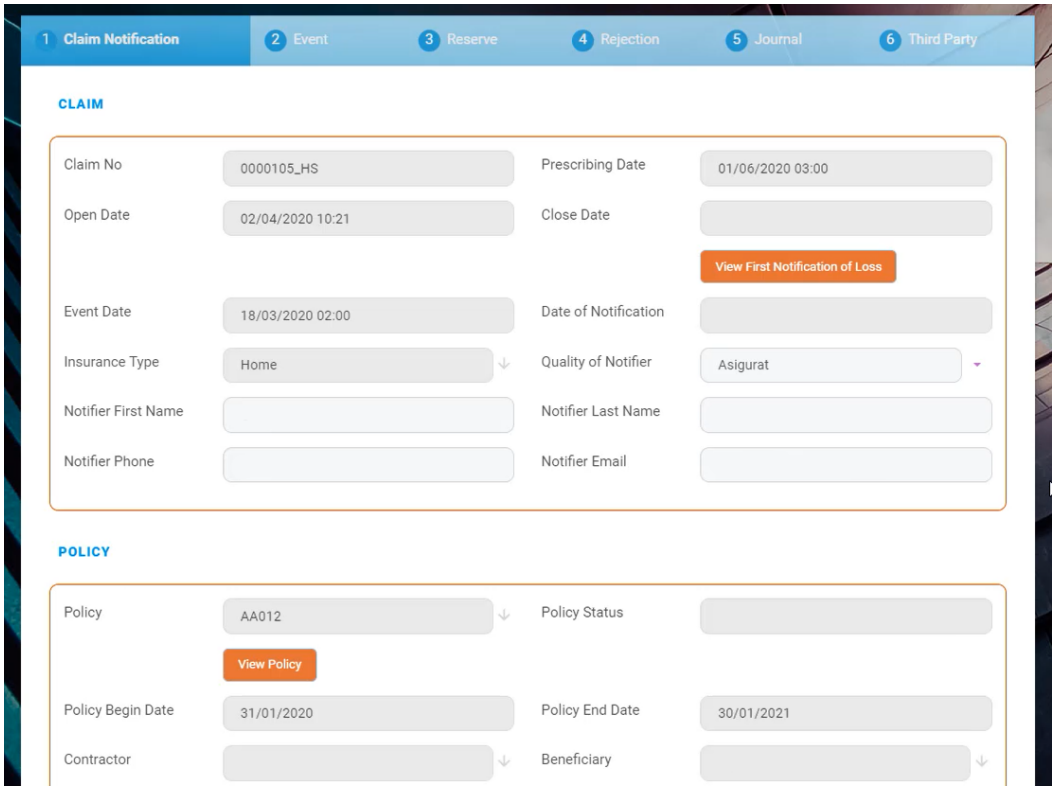
Create a Claim

FintechOS Core Claims Admin can be configured anytime after the FNOL is initiated. Go to the **FNOL list** page and select the desired FNOL. Go to **View Claim** to open the **Core Claims Admin** wizard and create a new claim record. The wizard has eight steps to help you complete the processing of the claim. Proceed with the tabs from left to right, in the following order:

Claim Notification

Go to the **Claim Notification** tab and click it. The **Claim Notification** window is displayed with the following sections: **Claim**, **Policy** and **Status**.

Check if there are any empty fields and complete those.



Below is a description of the three tables in the **Claim Notification** tab:

Field	Type	Description
Claim No	Text	It is read-only.
Open Date	Date time	It is read-only.
Event Date	Date Time	It is read-only.
Insurance Type	Option set	It is read-only.
Notifier First Name	Text	It is read-only.
Notifier Phone	Number	It is read-only.
Prescribing Date	Date Time	It is read-only.
Close Date	Date Time	It is read-only.
View First Notification Of Loss		It opens the FNOL.
Date of Notification	Date time	It is read-only.
Quality of Notifier	Option Set	It is read-only.
Notifier Last Name	Text	It is read-only.
Notifier Email	Text	It is read-only.

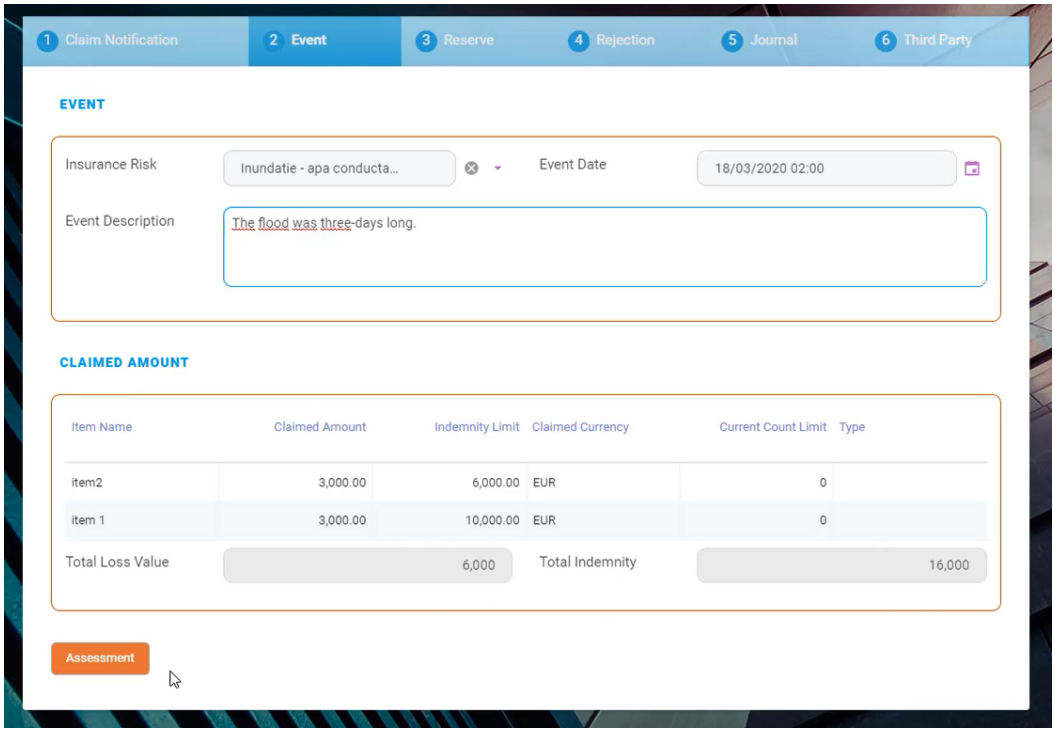
Field	Type	Description
Policy	Option set	It is read-only.
Policy Begin Date	Date	It is read-only.
Contractor	Option set	It is read-only.
Account Insured	Number	
Policy	Look up	
Policy Status	Look up	It is read-only.
Policy End Date	Date	It is read-only.
Beneficiary	Option set	It is read-only.
Owner	Option set	It is read-only.
Wording	File	

Field	Type	Description
Loss Value	Number	It is read-only.
Payments	Number	It is read-only.
Reserves	Number	It is read-only.
User	Option set	It is read-only. It is the account with which the user logged into the Portal.

Click **Save and reload** and proceed to the next tab.

Event

Click the **Event** tab. The **Event** window is displayed. Here you find the following sections: **Event** and **Claimed Amount** .



Check if there are any empty fields and complete those.

Below is a description of the two tables in the **Event** tab:

Field	Type	Description
Insurance Risk	Option set	It is read-only.
Event Date	Date time	It is read-only.
Event Description	Text Area	You can add more details here.

Field	Type	Description
Total Loss Value	Number	It is read-only.
Total Indemnity	Number	It is read-only.

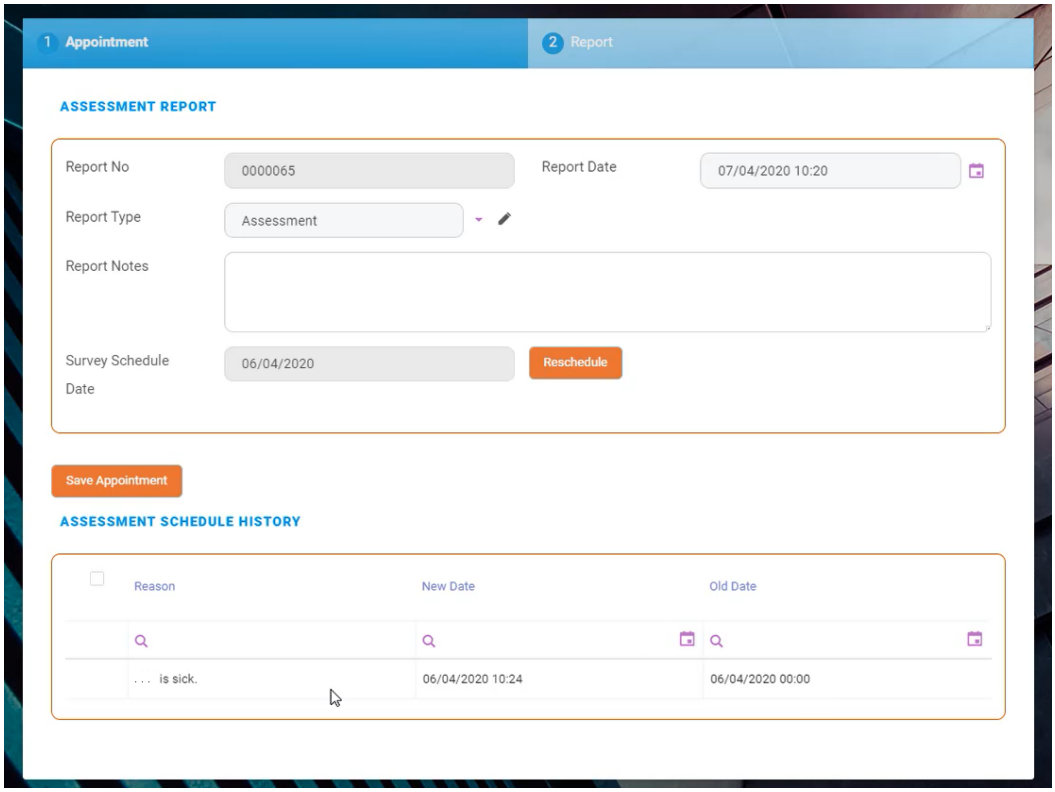
Click **Assessment** and proceed to the next tab.

Assessment

Click the **Assessment** tab. The **Assessment** window is displayed.

Click **Insert** to open a new assessment page.

Below you find the following sections: **Assessment Report** and **Assessment Schedule** .



Check if there are any empty fields and complete those.

Below is a description of the two tables in the **Assessment** tab:

Field	Type	Description
Report No	Text	The number of the registered report. It is obtained using sequencer functionality.
Report Date	Date time	It is the time when the report is done.
Report Type	Option set	It is the type of assessment. It can take the following values: Assessment or Self-Assessment.
Report Notes	Text area	Provide details that help asses the event.
Survey Schedule Date	Date time	It is mandatory to be filled in. It is the time when the assessment was done.

When you click **Save Appointment**, the appointment history becomes visible underneath.

On the **Report**section click **Insert** to upload the assessment report. If ever necessary, you can also erase a report, by pressing **Delete**.

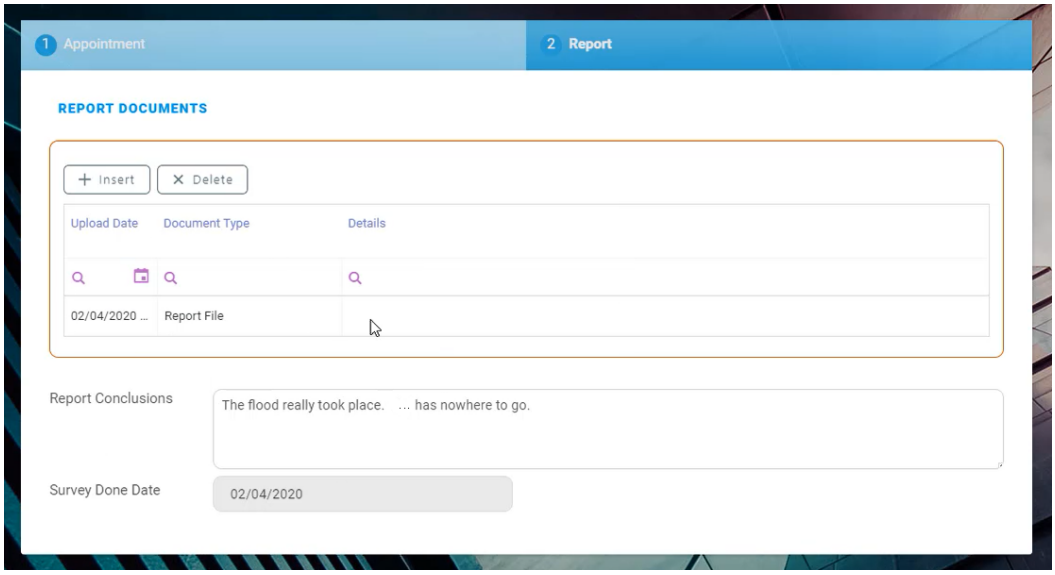
When you click **Insert**, a new table becomes visible.

Field	Type	Description
Upload Date	Date time	It is read-only. It is automatically filled.

Field	Type	Description
Document Type	Option set	It is mandatory. You can opt for uploading: Other Documents, Pictures, Report files or Invoices.
Files		Add the file here.
Details	Text area	Describe if necessary.

Click **Save and close**. The **Report Conclusions** and **Survey Done Date** fields become visible.

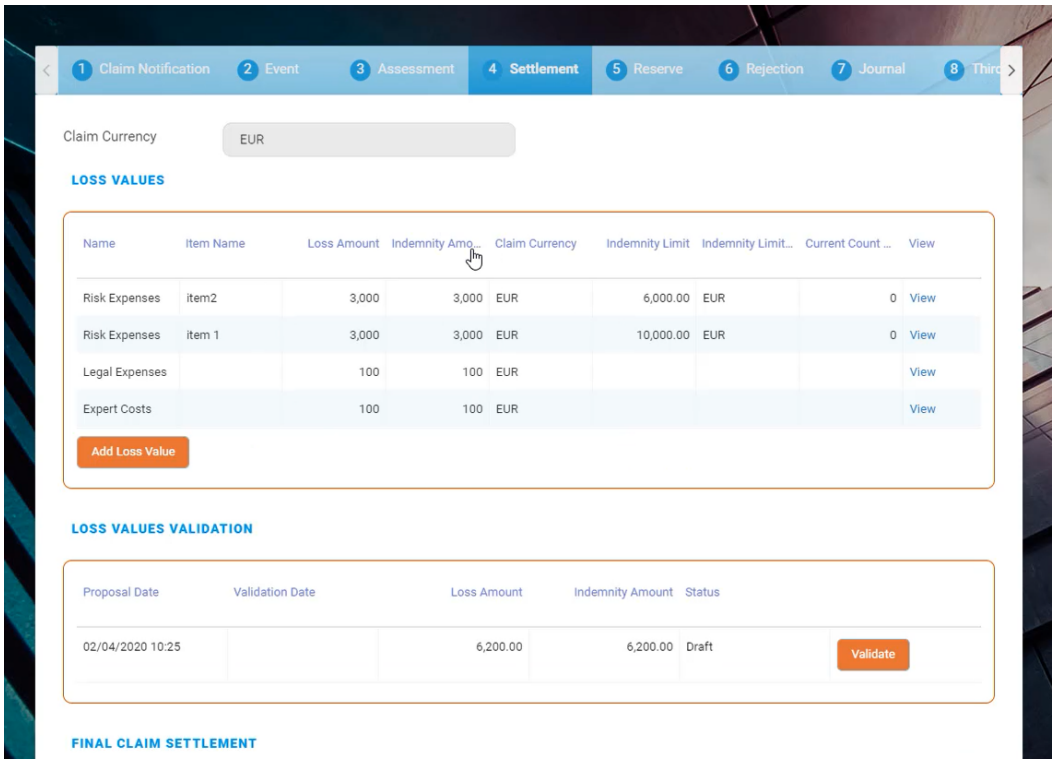
Complete the conclusions and the date.



Click **Close report**, click **Save and close** and then click **Settlement** to move to the next tab.

Settlement

Click the **Settlement** tab. The **Settlement** window is displayed with the following sections: **Loss Values**, **Loss Values Validation** and **Final Claim Settlement**.



Click **Add Loss Value**. The page loads again and you can insert the values. Choose the currency with which to pay the policy holder. Also, insert the amount of money for each item that the policy holder makes a claim for.

In the **Loss Values Validation** section, click **Validate** to launch the request for validation from a **Core Claims Admin** super user.

In the **Final Claim Settlement** section, click **Insert** to upload more files, when needed. You can upload the following types of files:

- Repair Quote
- Ownership Documents
- Other Documents
- Event Pictures

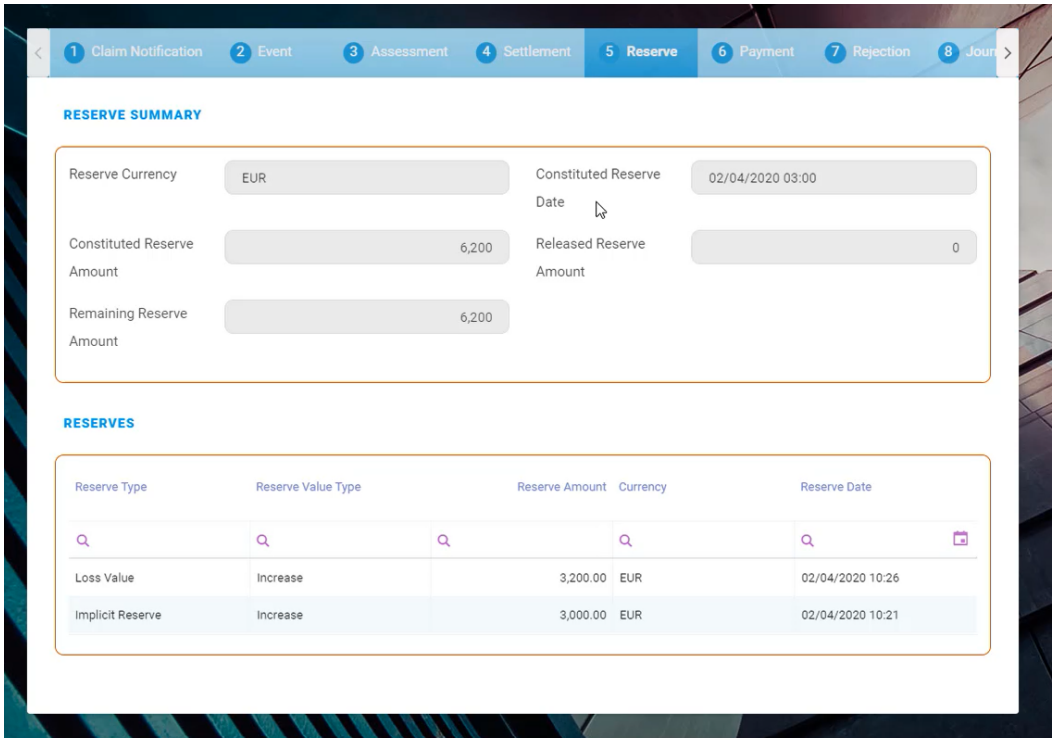
- Payment
- Invoices
- Authorities Documents
- ID Copy

After this step, click **Risk Covered** if you need to view the covered risk.

Click **Save and reload** and then click **Reserve** to go to the next tab.

Reserve

When you click **Reserve**, the **Reserve** window is displayed with the following sections: **Reserve Summary** and **Reserves**.



Below is a description of the two tables in the **Reserve** tab:

Field	Type	Description
Reserve Currency	Option Set	It is read-only. It is the currency of the reserve.
Constituted Reserve Amount	Number	It is read-only.
Remaining Reserve Amount	Number	It is read-only.
Constituted Reserve Date	Date time	It is read-only.
Released Reserve Amount	Number	It is read-only.

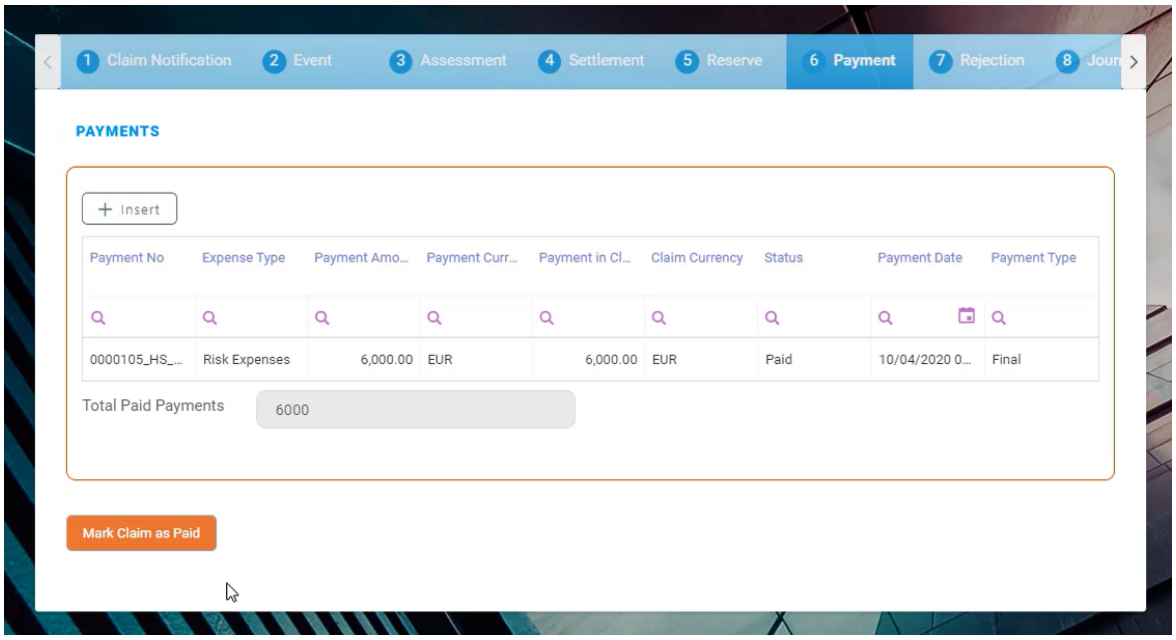
Field	Type	Description
Claim	Lookup	Read-only. The ID of the claim.
Reserve Date	Date time	Read-only. The date when the reserved amount is processed.
Reserve Type	Option Set	Read-only. The type of reserve agreed for the policy.
Reserve Value Type	Option Set	Read-only.
Currency	Lookup	Read-only. The currency of the reserve.

HINT

The next step of the **Core Claims Admin** digital journey is to create a **Payment Proposal**. Click Payment.

Create a Payment Proposal

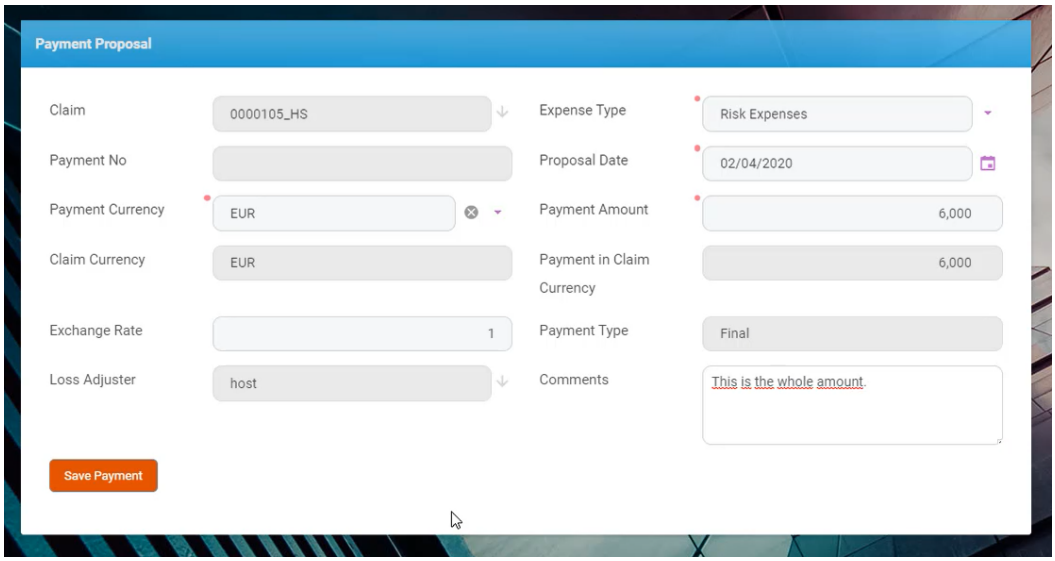
Payment proposals can be made anytime after the FNOL is initiated. They have a dedicated page in the **Core Claims Admin** digital journey. Click the **Payment** tab. The **Payment** window is displayed. You notice that some fields are already populated with data extracted from the policy.



Click **Insert** to add a new **Payment Proposal**. The Payment flow becomes active. This flow has four steps and helps you to manage the claim's payment proposals. Proceed from left to right, in the following order:

1. Payment proposal

Complete the fields from the **Payment proposal** table.



Below is a description of the table:

Field	Type	Description.
Claim	Option set	It is read-only. It is the claim number.
Payment No	Text	It is the number of the payment.
Payment Currency	Option set	It is the currency for the payment.
Claim Currency	Text	It is the currency for the claim.
Exchange Rate	Number	For inserting the exchange rate values.
Loss Adjuster	Option set	It is the person recording the Claim.
Expense Type	Options set	It is required to fill in. It is the type of expense for which this payment is done.

Field	Type	Description.
Proposal Date	Date	It is required to fill in.
Payment Amount	Number	It is required to fill in.
Payment in Claim Currency	Number	It is auto-filled.
Payment Type	Text	It is read-only. It can be: Final or Partial.
Comments	Text area	For any details regarding the payment.

Click **Save Payment**. A new step becomes available.

2. Approve Payment

Go to the **Approve payment** step. Complete the fields from the **Approve payment** table.

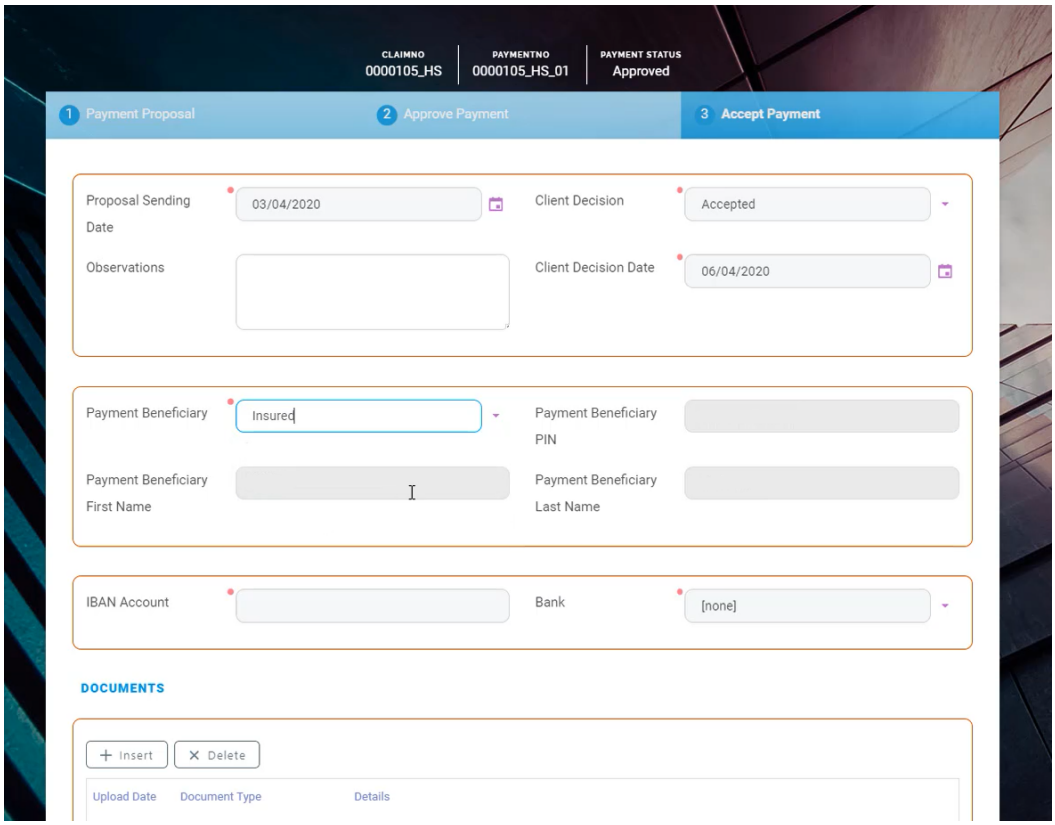
Below is a description of the table:

Field	Type	Description
Approval Date	Date time	It is the time of the approval . It is auto-filled.
Claim Manager	Option set	It is the person who makes the approval.
Observations	Text area	For any details regarding the approval.

Click **Approve** or **Unapprove**. For the **Unapprove** flow please see the details from the Business Statuses model.

3. Accept Payment

After you clicked **Approve**, the third step **Accept Payment** becomes available. Complete the fields from the **Accept Payment** table.



Below is a description of the table:

Field	Type	Description
Proposal Sending Date	Date	It is required to fill in.
Client Decision	Option set	It is required to fill in.
Observations	Text Area	Add any relevant observations.
Client Decision Date	Date	It is required to fill in.
Payment Beneficiary	Option set	The Loss Adjuster choose the beneficiary.
Payment Beneficiary First Name	Text	It is the name of the person.
Payment Beneficiary PIN	Text	It is the PIN of the person.

Field	Type	Description
Payment Beneficiary Last Name	Text	It is the name of the person.
IBAN Account	Text	The IBAN of the beneficiary.
Bank	Option set	Bank corresponding to the IBAN.

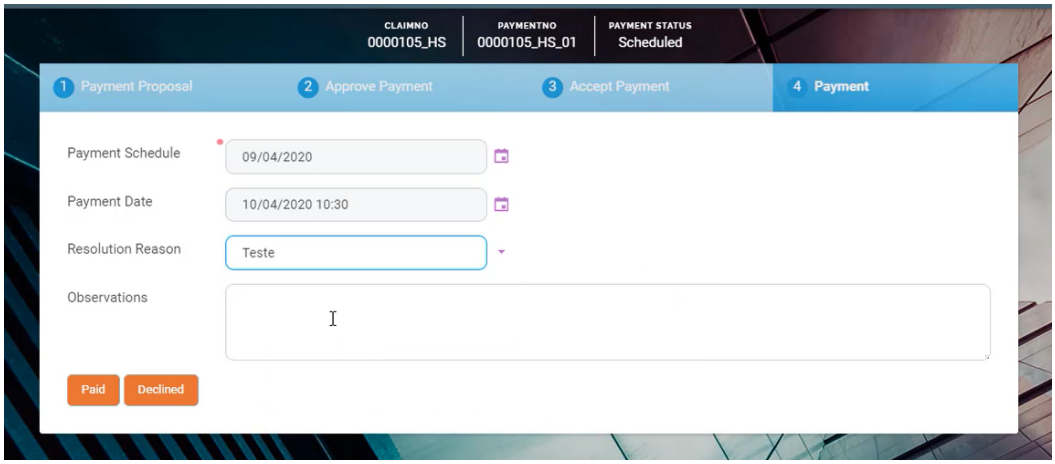
IMPORTANT!
Client Decision Date must be set after the **Proposal Sending Date**.

Under **Documents**, click **Insert** to add a payment request/ mandate/ other documents.

Click **Save Client Decision**.

4. Payment

Go to the **Payment** step. Next to **Payment Schedule**, click the **Calendar** icon to set the date and then click **Schedule**.



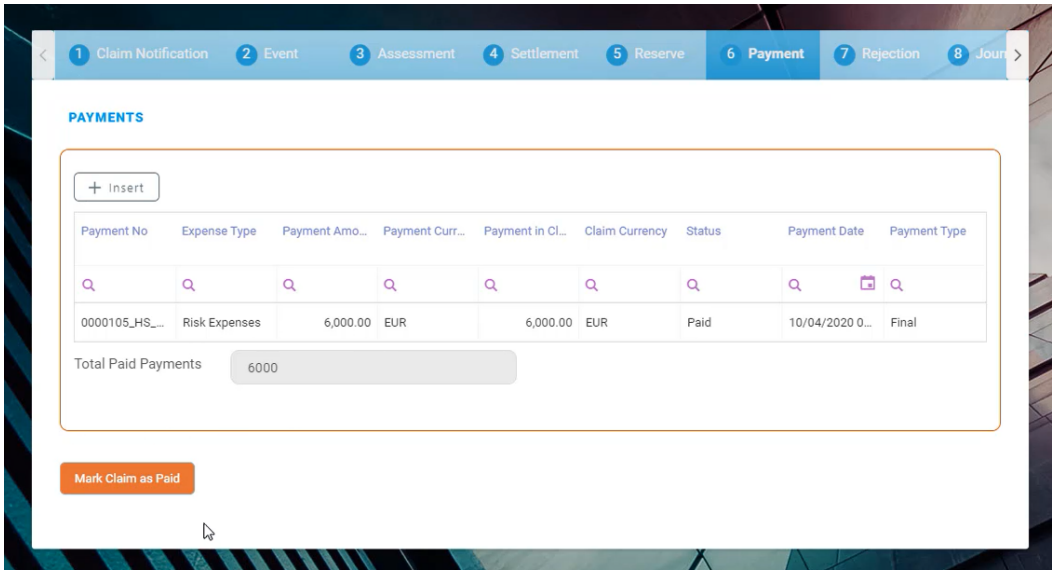
After clicking **Schedule**, three new fields become available: **Payment Date**, **Resolution Reason** and **Observations**.

Filed	Type	Description
Payment Schedule	Date	It is required.

Filed	Type	Description
Payment Date	Date time	It must be after the payment schedule.
Resolution Reason	Option set	
Observations	Text	For details regarding the resolution.

Click **Paid** or **Declined**.

After clicking **Paid**, click **Mark Claim as Paid** or click **Insert** to add new payments.



HINT

The next step of the **Core Claims Admin** digital journey is to manage any claim related information. Go to the Rejection, Journal & Third Party Details for more information about these journey steps.

Rejection, Journal & Third Party Details

These are the final steps of the **Core Claims Admin** digital journey. Use them to insert information about rejection or third party related to the claim, if the case, or to manage the claim documents.

Rejection

On the **Rejection** tab, you find the **Cancel** and **Decline** options.

Press **Cancel** when the client chooses to discard the claim. Press **Decline** when the insurance company chooses to discard the claim - for example the claim does not fall under the policy scope.

After pressing either of the buttons, the wizard stops - i.e. you cannot continue or return to the payment flow.

> **IMPORTANT!**
> If you press **Risk Covered** this step is not shown in your flow.

Journal

This is a section where you find all the documents about the claim.

Third Party

When needed, use this section to fill in information about a third party involved in the claim.

Field	Type	Description
TP Liable	Bool	Set to true if there is a third party liable.

Field	Type	Description
TP Observations	Text area	For details regarding the third party.

Libraries and Scripts

Described below are the tabs and the corresponding libraries and scripts are used for the Core Claims Admin solution.

Payment

The following scripts and libraries are used for the **Payment** tab:

Server Automation Script

The following script is used for the **Payment** Tab:

FTOS_CLAIM_ClaimPaid

This script calls the function claimPaid from the “FTOS_CLAIM_ ClaimPaid” (server automation script library).

Server Automation Script Library

The following scripts are used:

FTOS_CLAIM_ClaimPaid

This script contains the following functions:

`claimPaid();`: This function performs different validations, updates, inserts or businessStatus changes based on claimId and uses “FTOS.Claim” and “FTOS_INS_Utils” (server automation script libraries) as helpers.

Input parameters:

- claimId - (string) - the claim ID.
- claimPaidLibrary - (string) - DifferentValues(not used).

Output parameters:

- N/A.

`changePaymentStatus();`: This function performs a fetch based on the claimId and updates or changes the businessStatus based on the returned results of the fetch.

Input parameters:

- claimId - (string) - The claim ID.

Output parameters:

- N/A.

`getRiskClaimedId();`: This function performs a fetch based on the insuranceProductItemId and claimId to retrieve the RiskClaimedId.

Input parameters:

- insuranceProductItemId - (string) - the insurance product item ID.
- claimId - (string) - the claim ID.

Output parameters:

- riskClaimedId or null.

Rejection

The following scripts are used for the **Rejection** tab:

Server Automation Script

The following script is used for the **Rejection** tab:

FTOS_CLAIM_Assessment_DeclineCancelClaim

This script is called on the **Rejection** step of the FTOS_CLAIM_Claim → Claim form and receives an object (context.Data) with the following data:

- **claimId** = context.Data.claimId;
- **btn** = context.Data.btn;
- **remainingReserveAmount** = context.Data.remainingReserveAmount;

Based on the received data, the script runs an update, a fetch and some validations for an object creation in order to run the function setData to set data for the FTOS_Claim_DecliningProposal → DecliningProposals view.

Script update

If remainingReserveAmount is 0, the update for FTOS_CLAIM_Claim entity is triggered and changes the claimCloseDate attribute with the current date for the current claimId.

Script fetch

paymentList – This is a fetch for FTOS_CLAIM_Payment entity to get the attributes FTOS_CLAIM_Paymentid, businessStatusId and expenseTypeId based on the current claimId.

FTOS_CLAIM_Assessment_ChangeBusinessStatusPayments

This script changes the businessStatus of “FTOS_CLAIM_Payment” based on different conditions.

FTOS_CLAIM_ClaimChangeStatus

This script performs different validations, businessStatus changes, inserts and updates and uses “FTOS.Claim” and “FTOS_INS_Utils” (server automation script libraries) as helpers.

Third Party

The following scripts and libraries are used for the Third Party tab:

Server Automation Scripts

The following scripts are used for the **Third Party** tab.

FTOS_CLAIM_LossValueAdd

This script performs different fetches, inserts, updates and validations to insert LossValue for a new claim, based on the claimId. Also, this script uses the FTOS_INS_Exchange and the FTOS.Claim libraries.

FTOS_CLAIM_LossValueValidate

This script performs a fetch, different validations, an update to FTOS_CLAIM_LossValue, businessStatus changes and an insert to FTOS_CLAIM_Reserve. Also, this script uses the FTOS_INS_Utils and the FTOS.Claim libraries.

FTOS_CLAIM_ReserveAfterInsert

This script performs different fetches, updates on FTOS_CLAIM_Claim and validations, based on the claimId, reserveTypeId, reserveAmount and currencyId.

Server Automation Script Library

The following scripts are used:

FTOS_CLAIM_FNOL

This script library holds the following functions:

`getAccountByPhoneNumber();`: This function returns the client account data based on the phone number.

Input parameters:

- phoneNumber - (string) - The client phone number

Output parameters:

- An array with the client account info or null if there is no client with the queried phone number.

`getPoliciesByAccountId();`: This function returns the polices of a client.

Input parameters:

- accountId - (string) - The client account ID.
- type - (string) - The client type (i.e. “Contractant”)
- policyType - (string) - Policy type.

Output parameters:

- An array with the client policies.

`getPolicyByNumber();`: This function returns the policy by the policy number

Input parameters:

- policyNumber - (string) - The policy number.

Output parameters:

- An array with the policy data or null if there is no policy with queried number.

`updateClaimedRisk();`: This function updates the "FTOS_CLAIM_RiskClaimed" entity based on the FNOL ID.

Input parameters:

- FNOLId - (string) - The FNOL ID.
- updateData - (object) - An object with data.

Output parameters:

- N/A

`getFNOLDocuments();`: This function returns the FNOL documents based on the FNOL ID.

Input parameters:

- FNOLId - (string) - The FNOL ID.

Output parameters:

- An array with the FNOL documents data or an empty array if there are no documents uploaded for the queried FNOL ID.

`getClaimedRisksByFNOL();`: This function returns the Claimed risks based on the FNOL ID.

Input parameters:

- FNOLId - (string) - The FNOL ID.

Output parameters:

- An array with the Claimed risks data.

`getDataForClaimedType();`: This function returns different data for the Claimed type based on the FNOL ID.

Input parameters:

- FNOLId - (string) - The FNOL id.

Output parameters:

- An array with the Claimed type data.

`getClaimSettingsByCode();`: This function returns a flow parameter based on the flow parameter code.

Input parameters:

- code - (string) - The flow parameter code.

Output parameters:

- An array with the flow parameter or null if nothing found.

`getDaysSinceLastFNOL();`: This function returns the number of days since the last FNOL(if any).

Input parameters:

- policyId- (string) - The policy id.
- riskId - (string) - The risk id.

Output parameters:

- The number of days or 0 if nothing found.

`getDaysBetweenEventAndPolicy();`: This function returns the number of days between event and policy start date.

Input parameters:

- FNOLId - (string) - The FNOL ID.

Output parameters:

- The number of days or 0 if nothing found.

`getClaimedAmountPercent();`: This function calls the `getClaimedRisksByFNOL();` function described above and does some calculation to return the claimed amount percent.

Input parameters:

- FNOLId - (string) - The FNOL ID.

Output parameters:

- The claimed percent amount 0 if nothing found.

`getFNOLbyClaimId();`: This function returns the FNOL based on the claimed id.

Input parameters:

- claimId - (string) - The claimed ID.

Output parameters:

- An array with the FNOL or an empty array if nothing found.

`getPolicyByClaimId();`: This function returns the policy based on the claimed id.

Input parameters:

- claimId - (string) - The claimed ID.

Output parameters:

- An array with the policy or an empty array if nothing found.

`getFirstNotificationByEventReportId();`: This function returns the first notification based on the event report ID.

Input parameters:

- eventReportId - (string) - The event report ID.

Output parameters:

- An array with the first notification or an empty array if nothing found.

`getPolicyAddress();`: This function returns an object with the address for the queried policy based on the policy id.

Input parameters:

- policyId - (string) - The policy ID.

Output parameters:

- An object with the policy address.

`getInsuredItems();`: This function returns the insured items based on the policy id.

Input parameters:

- policyId - (string) - The policy ID.

Output parameters:

- An object with the insured items.

FTOS_CLAIM_Reserve

This script library contains a main function called `ClaimReserve` that contains the following functions:

`getReserve();`: This function returns the following attributes from the `FTOS_CLAIM_Reserve` entity based on the reserve id:

FTOS_CLAIM_Reserveid

- reserveAmount
- currencyId
- reserveValueTypeId
- reserveDate
- claimId

Input parameters:

- reserveId - (string) - The reserve ID.

Output parameters:

- An an object with the queried data.

`getClaimPolicy();`: This function returns the following attributes from the `FTOS_CLAIM_Claim` entity based on the claim ID:

- FTOS_CLAIM_Claimid
- policyId
- claimNo

- claimOpenDate
- Input parameters:
- claimId - (string) - The claim ID

Output parameters:

- An an object with the queried data.

`getLastReserveId();`: This function returns the last reserve ID with the **Constituted** business status from the `FTOS_CLAIM_Reserve` entity based on the claim ID:

Input parameters:

- claimId - (string) - The claim ID

Output parameters:

- The reserve ID or null.

`getReserveDetail();`: This function returns an array with the reserve details from the `FTOS_CLAIM_ReserveDetail` entity based on the reserve ID:

Input parameters:

- reserveId - (string) - The reserve id

Output parameters: an array with the following data:

- insuranceProductItemId
- lossValue
- expertCostsValue

- legalExpensesValue
- previousReserveId
- paymentValue

`getReserveType();`: This function returns the reserve type and reserve value type from the `FTOS_CLAIM_Reserve` entity based on the reserve ID:

Input parameters:

- reserveId - (string) - The reserve ID

Output parameters:

- An object with the reserve type and reserve type value or an empty object if nothing found.

`reserve_GL_Operation();`: This function inserts a new transaction with the functions presented above and with the help of other functions, based on the reserveId and collected data from the previously presented functions.

Input parameters:

- reserveId - (string) - The reserve ID.
- operationTransactionName - (string) - Not used.

Output parameters:

- N/A

`getReserveDetailTotal();`: This function calculates the sum of the loss value from the reserve detail

Input parameters:

- reserveDetail - (array) - An array with all the reserve details.

Output parameters:

- sum - (numeric) - The sum of the loss value.

`ClaimReserve();`: This function returns an object with the results of the functions presented above, except reserve_GL_Operation, which is not used anymore:

Input parameters:

- N/A

Output parameters: an object with the following data:

- getReserve
- getReserveDetail
- getLastReserveId
- getReserveDetailTotal
- getReserveType

On Demand Scripts

The following scripts are used for Core Claims Admin:

FTOS_CLAIM_ClaimOpenFirstNotification :

This script opens the First Notification of Loss for a given ID.

Input parameters:

- firstNotificationId - the ID of the `FTOS_CLAIM_FirstNotification` entity.

Output parameters:

- N/A.

FTOS_CLAIM_ClaimOpenPolicy:

This script opens the Claim associated to the Policy with given policy ID.

Input parameters:

- policyId - the ID of the `FTOS_INSPA_Policy` entity.

Output parameters:

- N/A.

FTOS_CLAIM_RecalculateIndemnityAmount:

This script alculates and updates the value for the indemnity amount on each risk claimed of the claim taking in consideration certain attribute values of the claim.

Input parameters:

- claimId - the ID of the `FTOS_CLAIM_Claim` entity.

Output parameters:

- N/A.

FTOS_CLAIM_ClaimChangeStatus:

This script changes the status of the claim and then adjusts the reserve, risks claimed and loss values according to the type of status transition.

Input parameters:

- claimId - the ID of the `FTOS_CLAIM_Claim` entity.
- newBusinessStatusName – the name of the status in which the claim will be changed to.

Output parameters:

- N/A.

FTOS_CLAIM_ClaimGetInsuredRisk:

This script modifies the values of the risks claimed taking into consideration the claim details and the insurance risk.

Input parameters:

- claimId - the ID of the `FTOS_CLAIM_Claim` entity.
- policyId – the ID of the `FTOS_INSPA_Policy` entity.
- fnolId – the ID of the `FTOS_CLAIM_FirstNotification` entity.
- insuranceRisk – the ID of the `FTOS_IP_InsuranceRisk` entity.

Output parameters:

- N/A.

FTOS_CLAIM_DecliningProposal_AfterReopenClaim:

This script cancels the declining proposals for the given ID claim.

Input parameters:

- claimId - the ID of the `FTOS_CLAIM_Claim` entity.
- newBusinessStatusName – name of the status in which the claim will be changed to.

Output parameters:

- N/A.

FNOL API

Server Automation Script

The following server automation script is used:

FTOS_FNOL_API_Call

This script is called with an object as data and calls the insertFnol function from the FTOS_FNOL_API server automation script library.

Server Automation Script Library

The server automation script library contains the following script:

FTOS_FNOL_API

This script creates a new FNOL (first notification of loss) based on the object received from the FTOS_FNOL_API_Call script. The script contains the following functions:

getNotifier();

This function fetches (with fluentQuery) the customer's personal info based on the the personal numeric number CNP or PIN, only if the customer exists in database.

Input parameters:

- cnp -- (string) -- The customer's PIN/CNP.

Output parameters:

- The customer info or null if the customer does not exists in the database.

getTypeId();

This function returns the ID of the person type based on the personType received from the API call.

Input parameters:

- typeString –(string)- the person type (example: “Individual person”).

Output parameters:

- The ID of the person type.

getInsuredRisk();

This function checks if the customer is insured for the claimed risk.

Input parameters:

- policyId –(string)- The policy Id
- ojbInsuranceRisk –(string)- The name of the risk (example: Aircraft fall);

Output parameters:

- Insured risk ID and name, or null if the risk is not covered.

getRiskClaimed();

This function returns the policy item name, if the name exists in the database.

Input parameters:

- riskName – (string) – The name of the policy item
- insuranceRiskId – (string) – The risk ID returned from the getInsuredRisk(); function
- policyId – (string) – The policy ID.

Output parameters:

Returns an object with the following data, or null if there are no entries to match the query:

- riskCurrencyId from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.
- currentValueLimit from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.
- currentCountLimit from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.
- FTOS_INSPA_PolicyInsItemXCoveredRiskid from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.
- policyInsuranceItemId from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.
- insuranceRiskId from `FTOS_INSPA_PolicyInsItemXCoveredRisk`.

- FTOS_INSPA_PolicyInsuranceItemid from `FTOS_INSPA_PolicyInsuranceItem`.
- insuredObjectId from `FTOS_INSPA_PolicyInsuranceItem`.
- name from `FTOS_INSPA_PolicyInsuranceItem`.

dateVerify();

This function verifies that the given date is before or the current date.

Input parameters:

- selectedDate – (string) – The date.

Output parameters:

- Returns the given date, or false if the date is in the future.

checkPolicyVailidity();

This function checks if there is any policy with the given id.

Input parameters:

- policyId – (string) – The policyid.

Output parameters:

An object with the following data, or null if there is no policy with the given ID.

- FTOS_INSPA_Policyid from FTOS_INSPA_Policy
- policyNo from FTOS_INSPA_Policy

- issuedDate from FTOS_INSPA_Policy
- policyBeginDate from FTOS_INSPA_Policy
- policyEndDate from FTOS_INSPA_Policy
- businessStatusId from FTOS_INSPA_Policy
- insuranceTypeId from FTOS_INSPA_Policy
- beneficiaryId from FTOS_INSPA_Policy
- accountId from FTOS_INSPA_Policy.

getInsuredObjectAddress();

This function returns the address of the insured object, based on the given policy ID.

Input parameters:

- policyId – (string) – The policy id.

Output parameters:

Returns an object with the following data, or null if there are no entries for the given policy ID.

- FTOS_INSPA_Policyid from FTOS_INSPA_Policy
- accountId from FTOS_INSPA_Policy
- insuredId from FTOS_INSPA_Policy
- insuranceTypeId from FTOS_INSPA_Policy
- beneficiaryId from FTOS_INSPA_Policy
- wording from FTOS_INSPA_Policy

- policyPDF from FTOS_INSPA_Policy
- streetNo from FTOS_INSQB_Address
- postalCode from FTOS_INSQB_Address
- buildingNo from FTOS_INSQB_Address
- floorNo from FTOS_INSQB_Address
- apartmentNo from FTOS_INSQB_Address
- streetName from FTOS_INSQB_Address
- istrictId from District
- cityId from City.

insertFnol();

This function inserts a new FNOL based on the given object from the `FTOS_FNOL_API_Call` server automation script and uses all the functions presented above the fetch different data and to make different validations and also uses the following function from FTOS_INS_Utils:

getIdByAttrib();

This function returns the id of the given attribute.

Input parameters:

- entityName – (string) – The entity name;
- searchAttribute – (string) – The name of the attribute;
- searchValue – (string) – The attribute value.

Output parameters:

- The ID of the searched value, or null if there are no entries with the searched value.

Beside inserting a new FNOL, this function also inserts the following:

- Inserts a new customer into the Account entity if the customer does not exist, validation based on the given CNP/PIN;
- Inserts item/s into FTOS_CLAIM_AffectedItem;
- Inserts the claimed amount for the claimed risk into FTOS_CLAIM_RiskClaimed;
- Inserts document/s into FTOS_CLAIM_Document.

Input parameters:

- Token – (object) – The object received from the FTOS_FNOL_API_Call (server automation script)

Output parameters:

- The id of the new inserted FNOL

Glossary

C

Claim prescribing date

The time limit for making a claim under an insurance policy.

F

First notice of loss (FNOL)

The first notice of loss (FONL) is the initial report made to an insurance provider following loss, theft, or damage of an insured asset. The first notice of loss (FNOL), also known as the first notification of loss, is normally the first step in the formal claims process lifecycle.

L

Loss adjuster

An insurance agent who assesses the amount of compensation that should be paid after a person made a claim on their insurance policy and recommends settlement options based on estimates of damage and insurance policies held.

R

Reserve

The part of the reserve of an insurance company to be absorbed from the initial reserve in any year in payment of losses.

S

Settlements

Making a settlement with the insurance company means that both the insured client and the insurance company agree to a payment amount, which is almost always less than the amount the client originally claimed.

T

TP Liable

Third-party insurance is an insurance policy purchased for protection against the claims of another.