A woman with dark hair is shown in profile, looking down at a tablet she is holding. The background is a blurred city skyline. Overlaid on the lower left of the image are semi-transparent financial charts, including a line graph with orange markers and a bar chart.

Proposal Configurator 1.4.0

User Guide

TOC

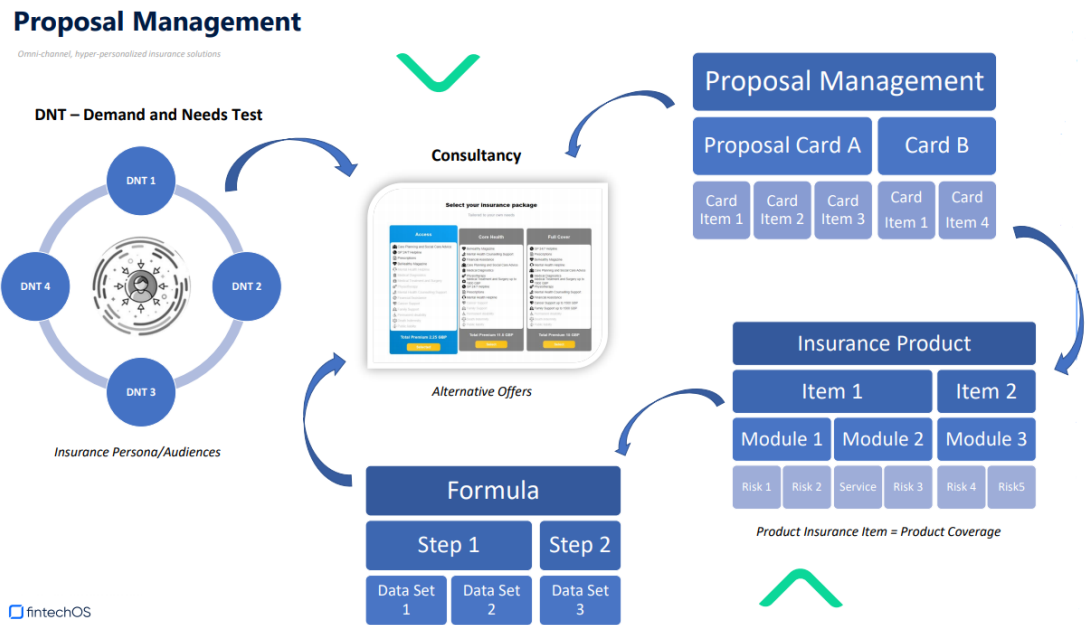
Overview	3
Install Proposal Configurator	4
Prerequisites	4
Installation Steps	4
Post-Installation Setup	6
Prerequisites	6
Installation Steps	7
Upgrade	9
Post-Installation Setup	10
Quote and Bind Configuration	11
Configure the Quote	15
Configure DNT Questions	17
Configure the Audience	19
Configure Card Items	19
Configure Cards	21
Configure DNT Question Answers	27
Configure the Insurance Persona	29
Proposal Configurations	32
Proposal Configurator API	32
Proposal Configurator Endpoints	68
Proposal Configurator Scheduled Jobs	85

Overview

Proposal Configurator offers the ability to configure product offerings in logical bundles, covering core and optional benefits comparatively priced by level and benefit type.

Proposal Configurator complies with the Insurance Distribution Directive (IDD) and Demands and Needs Test (DNT) requirements, to deliver customized insurance propositions to customers, with clarity around breadth of coverage and the cost associated with it. The solution allows system operators to organize data for the Quote&Buy process and insurance products.

Learn about the solution's benefits and how to configure your offer bundles by accessing the [Quote and Bind Configuration](#) page.



Install Proposal Configurator

Proposal Configurator 1.5.0 is a part of the **Insurance Launch 4.6.0** package. Follow the instructions below to install and configure **Insurance Launch 4.6.0** if it's not already installed on your environment. This is the first package to be installed in the entire insurance suite.

Install Insurance Launch 4.6.0

Prerequisites

Before installing or upgrading to **Insurance Launch 4.6.0**, make sure the following are already installed:

- **FintechOS Platform v22.1.4**
- **SySDigitalSolutionPackages v22.1.4000.zip**

Installation Steps

1. Download the solution package from Release Hub.
2. Unzip your solution package .zip archive file. Identify the **01 DeploymentPackages** folder.
3. Locate the **FtosSysPkgDeployer** folder in the FintechOS installation kit (the path is <unzipped_install_archive>\Tools\FtosSysPkgDeployer). You need it to install the SySDigitalSolutionPackages.

4. Select and copy the **FtosSysPkgDeployer** folder next to the **01 DeploymentPackages** folder.
5. Create the .bat file needed for installation and save it next to the FtosSysPkgDeployer folder. Add the following script in the .bat file.

```
CD /D %~dp0
"%~dp0\FtosSysPkgDeployer\FtosSysPkgDeployer.exe" -i
-a -s "StudioLink" -u AdminStudioUser -p user_
password -z DataBaseServer -v DB_user -k DB_user_
password -d "TheNameOfTheDataBase" -r "%~dp0\01
DeploymentPackages\*.zip"
Pause
```

6. Execute the .bat file to start the installation.

.bat file script parameters

- <StudioLink> - The web URL of the Innovation Studio installation, for example http://localhost/ftos_studio.
- <AdminStudioUser> - 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.<user_password> - The password for the Innovation Studio user.
- <DataBaseServer> - 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.
- <TheNameOfTheDataBase> - The name of the database.
- <syspack_path>- The physical path to the unzipped Core Policy Admin v3.4.0 previously downloaded.
- <DB_user_password>- The password for the above mentioned SQL user.

Post-Installation Setup

After installing, perform the following configurations:

Modify the app-settings Studio Vault key by adding the .xls value to the FileUploadWhiteList key:

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

IMPORTANT!

If you decide to work with our demo products settings, download the **Proposal Configurator Import 1.5.0.zip** file and follow the same installation steps as for **Insurance Launch 4.6.0**.

Proposal Configurator Import 1.5.0.zip can be used only for first time imports. If you're already using a previous version of **Proposal Configurator Import** do not upgrade to this version.

Install Proposal Configurator 1.4.0

Prerequisites

Before installing or upgrading to Proposal Configurator **v1.4.0**, make sure the following are already installed:

- **FintechOS Platform v22.1.1.0**
- **SySDigitalSolutionPackages v22.1.0003**
- **Insurance Product Factory V4.5.0.**

Installation Steps

1. Unzip your **Proposal Configurator 1.4.0.zip** archive file.
2. Create a folder and name it **01 DeploymentPackages**.
3. Copy the **DigitalAsset** folder (if it exists) and **DigitalSolution M.N.U.P.RC.zip** file (extracted at step 1) inside the **01 DeploymentPackages** folder.
4. Locate the **FtosSysPkgDeployer** folder in the FintechOS installation kit (the path is <unzipped_install_archive>\Tools\FtosSysPkgDeployer). You need it to install the SySDigitalSolutionPackages.
5. Select and copy the **FtosSysPkgDeployer** folder.
6. Create the **install_Syspack.bat** file needed for installation.
7. Add the following script in the file and save it next to the **FtosSysPkgDeployer** folder.

```
CD /D %~dp0
"%~dp0\FtosSysPkgDeployer\FtosSysPkgDeployer.exe" -i
-a -s "StudioLink" -u AdminStudioUser -p user_
password -z DataBaseServer -v DB_user -k DB_user_
password -d "TheNameOfTheDataBase" -r "%~dp0\01
DeploymentPackages\*.zip"
Pause
```

8. Run the **install_Syspack.bat** file by double clicking on it.
9. Follow the steps above to import and install the **Proposal Configurator Import 1.4.0.zip**, if needed.

IMPORTANT!

The **Proposal Configurator Import v1.4.0** contains proposal configurator demonstration settings, adding the Personal

Accidents quote config settings. If you decide to work with our demo settings, install the **Proposal Configurator Import 1.4.0.zip**, following the standard installation steps **1-8**.

The **Proposal Configurator Import v1.4.0** solution has the following dependency: **Quote Property V1.1.0**. The **Quote Property V1.1.0** must be installed before, following the standard installation steps **1-8**, and then running the SQL scripts from **FTOS_INSQB_QuoteProperty** folder. Post-installation, copy the **a-check.svg** picture in FintechOS Portal → **Custom** → **img** folder.

10. After installing the **Proposal Configurator 1.4.0**, respectively **Proposal Configurator Import 1.4.0** digital solution, run the SQL scripts from the **Proposal Configurator**, respectively **Proposal Configurator Import** folder.

install_Syspack.bat file script parameters

- <StudioLink> - The web URL of the FintechOS Studio installation, for example http://localhost/ftos_studio.
- <AdminStudioUser> - The username of the FintechOS Studio user under which this import is executed. The user has to exist in FintechOS Studio prior to this operation.
- <user_password> - The password for the FintechOS Studio user.
- <DataBaseServer> - 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.
- <TheNameOfTheDataBase> - The name of the database where the Proposal Configurator **v1.4.0** is deployed.

- <syspack_path> - The physical path to the unzipped Proposal Configurator **v1.4.0** previously downloaded.
- <DB_user_password> - The password for the above mentioned SQL user.

Upgrade

If you already have the **Proposal Configurator Import v1.3.0** package installed, upgrade to **v1.4.0** by performing the below steps:

1. Check if near the **01 Deployment Packages** folder the **Proposal Configurator Import Upgrade** folder exists.
2. Run the SQL scripts located in the **Upgrade** folder.
3. Perform the [installation steps 1- 9](#) above, to install **Proposal Configurator v.1.4.0**.
4. Perform the same to install **Proposal Configurator Import Upgrade v1.4.0**.
5. After installing the **Proposal Configurator Import Upgrade v1.4.0** digital solution, perform the post-installation setup listed below.

IMPORTANT!

The **Proposal Configurator Import Upgrade v1.4.0** solution has the following dependency: **Quote Property V1.1.0**. The **Quote Property V1.1.0** package must be installed before, following the standard installation steps **1-8**, and then running the SQL scripts from **FTOS_INSQB_QuoteProperty** folder. Post-installation, copy the **a-check.svg** picture in FintechOS Portal → **Custom** → **img** folder.

Post-Installation Setup

After installing the **Proposal Configurator Import v1.4.0** digital solution, perform the following business configurations:

- The cards included in the packages are imported in Draft status. Change their statuses to Approved, in order to start using them in the business flows.

After installing the **Proposal Configurator Import Upgrade v1.4.0** digital solution, perform the following business configurations:

- Change the statuses the status of the cards previously imported using **Proposal Configurator Import V1.3.0** to Approved, in order to start using them in the business flows.

Quote and Bind Configuration

Proposal Configurator is used to display in a Quote&Bind application different offer options mainly based on the user's answers to DNT questions. So, using the answer and the proper configuration made through the **Proposal Configurator** solution, the system is able to create and expose suitable custom insurance offers for the client's needs.

For example, if there are 3 DNT answers to a specific DNT question such as:

- Option 1 (e.g. Home Insurance);
- Option 2 (e.g. Home Assistance);
- Option 3 (e.g. Civil Liability).

The user is able to select different combinations in the answering process:

- Select only Option 1;
- Select only Option 2;
- Select only Option 3;
- Select Option 1 and Option 2;
- Select Option 1 and Option 3;
- Select Option 2 and Option 3;
- Select all three options.

The DNT questions and answers are closely related to the configuration of an insurance product insofar as the chosen options are mapped to the items of the insurance products used for the offering process.

Therefore, based on what the user has chosen, the mapping is made between the all three options displayed in the Quote&Bind application and three correlated insurance items of the used insurance products:

- Insurance Item 1 (e.g. Home Insurance);
- Insurance Item 2 (e.g. Home Assistance);
- Insurance Item 3 (e.g. Civil Liability).

Utilize **Proposal Configurator** in conjunction with dynamic insurance customer journeys to:

- Provide the ability to set up the Demands and Needs (DNT) questions and answers be configured to the audiences. **Proposal Configurator** offers the possibility to adopt digitalization to generate aligned and suitable offers to meet customer needs;
- Present benefits, options and costs in a clear way for customers, and is Insurance Distribution Directive (IDD) compliant;
- Configure the proposal. Combine benefits to present multiple and comparable offer cards, assisting the customer to select the best product mix and benefit levels to fit their profile;
- Create different versions for your product.

The **Quote&Bind Configurator** is structured in components which represent the steps of configuration for a **Proposal Configurator** solution.

In **FintechOS Studio**, each component is marked as a **Q&B Configurator** sub-item menu:

- **Quote Config**: This represents the configuration of the Quote&Bind item which is integrated in the Quote&Bind journey application;
- **DNT Questions**: This represents the options to choose from in the Quote&Bind application;
- **Audience**: This contains the configured audiences representing the group of people to whom the bidding options are addressed;

- **Card Items:** This stores the configured Insurance Product Items which are included in the Quote&Bind application offers through which the Premium Amount calculations are performed;
- **Cards:** This represents the offers which are presented to the client in the Quote&Bind application;
- **DNT Question Answers:** This stores the answers to the options that a user can have, mainly yes or no answers;
- **Insurance Persona:** This represents the segments correlated to the audiences created. More precisely, an insurance persona must be defined for each created audience.

In order to properly configure a **Proposal Configurator** solution, take into consideration the following steps:

1. **Insert and Manage Cards and Card Items:** Insert multiple cards (offers) e.g. A (Base), B (Premium), C (Full), then insert each insurance product item as a card item. For each card, insert card Items, check if the options is active or not and set the order.
2. **Insert DNT Questions and Answers:** These are the options to be selected by the user in the Quote&Bind application, they then translate to boolean answers for the configured DNT question.
3. **Insert Quote Config:** Create a quote type for the desired Quote&Bind application to which it is correlated, and insert the available DNT questions.
4. **Insert the Audience Segment and the Audience:** Define an audience segment based on several criteria. E.g. A “Home Insurance” audience segment can be defined, containing the records in the Quote entity with the `answerId` attribute equal to “Yes - Home Insurance”, and create an audience based on one or more audience segments. See more details about audience and audience segments in the [Campaign](#) documentation.
5. **Insurance Persona and the Insurance Persona Card:** For each audience, define an insurance persona. For each insurance persona, add a number of the available cards, which represent the offer shown for that selection. Finally, add the insurance persona on quote configuration.

The following sections highlight benefits and features of creating your custom offer cards and defining the audience with the **Proposal Configurator** solution.

Compliant with Insurance Distribution Directive (IDD)

Proposal Configurator enables the dynamic assembly of propositions to be presented to those seeking insurance coverage, whether general insurance or life protection, at levels that can readily be compared and considered from the point of view of relevance and affordability. These propositions are formulated and presented at key stages in a FintechOS insurance customer journey, and can be stored and retrieved as required. They contribute to the interactive and inclusive user experience that cannot be delivered via the first-generation portal technologies.

It is therefore at this level that an insurer, managing general agent (MGA) or other form of distributor is able to differentiate itself from their competitors whilst satisfying IDD requirements of working within a level playing field from a product perspective.

Demands and Needs Test (DNT)

By completing the Demands and Needs Test (DNT), the customer provides personal and financial details that are pertinent to the lines of business they are inquiring into, including:

- Personal details relating to all family members and dependents;
- Financial details, including income, liabilities, outgoings, assets, etc;
- Existing personal insurance and employee benefit levels - life, health and pensions.

The information gathered through DNT is used to formulate coverage options that are aligned to the individuals' stated needs. It offers the benefit of gathering information and presenting insurance coverage options in a way that is:

- Accurate, personalised, suitable;
- Accessible, fair, inclusive;
- Affordable;
- Compliant.

Product Versioning

Product versioning helps to differentiate the sold product at any point in time, ensuring that customers can only buy the current version of a product and make the performance analysis easier. Insurance companies constantly monitor the performance of products to ensure that they can generate the required level of profit which is the difference between net premium collected (income) and the cost of claims paid (expenses). Underwriters can adjust premium rates and coverage to maintain a balance between profitability and price competitiveness.

When creating a new version of a product that is part of , the business user can replace the used formula with a new one. As soon as the new version of the product is approved, the user can see the premium amount calculated for each card using the new formula.

New versions of formulas can also be created and attached either on a coverage or on a product, that is included in the Proposal Configurator solution. When calculating the premium amount for each card, the user can see the result considering the result of the latest approved version of the formula. This can help underwriters with resetting the rates (pricing level) for a risk based on the underlying frequency and cost of claims.

Configure the Quote

In order to correlate the existing configuration with a quote process, you need to configure a quote registration. In this way, a quote configuration is integrated with the Quote&Bind application. To set up a quote configuration, you must provide the general information also correlate them with the previously configured [DNT question](#).

In this section, you can configure the quote types to be displayed on the offer cards.

1. Log in to **FintechOS Studio**.
2. Go to **Main Menu > Product Factory> Proposal Configurator > Quote Config**.
3. Click **Insert** to add a new quote.

4. Fill out the fields with the needed information.

- **Name:** The name of the quote, e.g. Bank Insurance;
- **Code:** Assign a code for the quote to make it easily recognizable;
- **Insurance Type:** Choose the type of insurance from the option set. Pre-configure the insurance type as per the steps detailed [here](#);
- **Description:** Free text field to describe the created quote;
- **Technical Documentation:** Input a technical description for the created quote, if needed;
- **Offer Template:** Upload a file representing an offer template to be printed.

5. Click **Save and Reload**. The **DNT Questions** and **Insurance Persona** grids are unfolded.

6. In the **DNT Questions** grid, create the questionnaire by adding the questions which are configured in the **Product Factory > Proposal Configurator > DNT Questions** section.

DNT Questions

[+ Insert existing](#) [X Remove existing](#)

Name	Question Type	Code	Order
Asigurarea obligatorie de locuinta (PAO)	Bool	PAO	1
Asigurarea extinsa a locuintei	Bool	ASEXT	2
Asistentia la domiciliu	Bool	ASD	3
Asigurarea bunurilor	Bool	AB	4
Raspundere civila	Bool	RC	5

7. Similarly, in the **Insurance Persona** grid, add the persona types which are configured in the **Product Factory > Proposal Configurator > Insurance Persona** section.

Insurance Persona

<input type="checkbox"/>	Name	Persona Type	Persona	Audience
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	Asigurare Baza	Audience		Asigurare Baza
	Asigurare Extinsa	Audience		Asigurare Extinsa
	Asigurare PAD	Audience		Asigurare PAD
	Asigurare Premium	Audience		Asigurare Premium
	Asigurare Totala	Audience		Asigurare Totala
	New test	Audience		New test

8. After configuring the **DNT Questions** and **Insurance Persona**, click **Save and Close**. The quote is listed in the **Quote Config List** grid.

Quote Config list

<input type="checkbox"/>	Name	Insurance Type	Code
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	Bank Insurance	Home	BRD
	Household Insurance	Home	HHI
	Home Insurance	Home	BNI

In **Quote Config**, you are also able to configure the **Main Product** which is used in order to give the renewal logic on a Masterpolicy, using the main product configurations, except for **Renewal Type** value. The **Main Product** attribute is displayed on the **Quote Config** form, which allows you to select an existing approved insurance product as being the main product in a quotation process and on a Masterpolicy.

Select the desired product from the list, and the renewal logic which is given, which is applied to a Masterpolicy, taking into consideration the main product's renewal configurations from product level, except the **Renewal Type**.

Configure DNT Questions

In this section, you can configure the DNT Questions that are further added in the **DNT Questions** grid when configuring the quote. These are then displayed in the application, and the offer is customized according to the customer's answer to the demands and needs test questionnaire.

1. Go to **Main Menu > Product Factory > Proposal Configurator > DNT Questions**. The DNT questions are listed in this screen, **DNT Questions List**.
2. Click **Insert** to add a new question. The following screen is displayed.

The screenshot shows a form titled 'DNT Question'. At the top right are three buttons: '+ Insert question', '+ Save and close', and '+ Save and close'. The form fields are: Name (Mandatory Home Insurance), Code (H0), Question Type (Bool), Description (Mandatory Home Insurance (H0)), Technical Description, and Tooltip (The H0 Mandatory Home Insurance covers the following risks: earthquakes, accidents, floods, damages to the building due to natural hazards).

3. Fill out the displayed fields:
 - **Name:** The DNT question name to be displayed in the Quote&Bind application;
 - **Code:** Assign a code to the DNT question;
 - **Question Type:** From the drop down, select the bool type of the question used in order to define the implied audiences;
 - **Description:** Free text field to input a description for the created question;
 - **Technical Description:** Input a technical description for the created question, if needed;
 - **Tooltip:** This field holds the description of the DNT Question from a customer point of view, that appears when the user hovers over the question.
4. After following these steps, click **Save and Close**. The question is now available in the **DNT Questions List** screen.

The screenshot shows the 'DNT Questions list' screen. At the top right are four buttons: '+ Insert', '+ Delete', '+ Export', and 'Q Advanced find'. Below the buttons is a table with the following data:

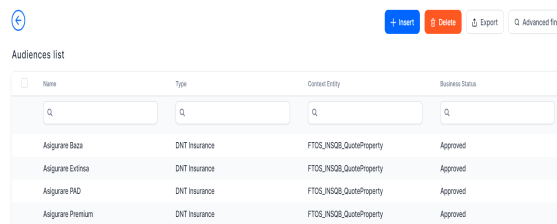
Name	Question Type	Code	Order
Time Limited	Bool	PET_LIM	31
Lifetime	Bool	PET_LFT	30
Online Channel	Bool	PET_ONL	29
Direct Channel	Bool	PET_DIR	28

Configure the Audience

The audience configuration process represents one of the major actions of customizing a Quote&Bind offer application. In this way, the offered insurance options can target a certain segment of customers so that the offers fit the needs of this created audience. The audience is configured using the [Hyperpersonalization Processor](#).

For the audience configuration process, you must firstly create targeted customer segments for the sales process, and then the created segments can constitute a customer audience to which insurers can address with various offers, in our case, with one or more offer cards.

1. Go to **Main Menu > Product Factory > Proposal Configurator > Audience**. The audiences list is displayed, as shown below.



Name	Type	Context Entry	Business Status
Audience Baza	DNT Insurance	FTOL_INSOB_QuoteProperty	Approved
Audience Ectha	DNT Insurance	FTOL_INSOB_QuoteProperty	Approved
Audience PAD	DNT Insurance	FTOL_INSOB_QuoteProperty	Approved
Audience Premium	DNT Insurance	FTOL_INSOB_QuoteProperty	Approved

2. Add new audience types as per the [Hyperpersonalization Processor](#) documentation.

Configure Card Items

In order to combine the Quote&Bind process with the coverages of the product, you need to configure the card items, that are then linked with the product item. Then, the offer types are created. You can choose the offers to display to a specific type of person and the number of cards.

Different versions can be created for the cards, by clicking Insert in the **Card Item Versions** section, and inputting the needed details.

1. Go to **Main Menu > Product Factory > Proposal Configurator > Card Items**.
2. Click the **Insert** to add a new card. The following screen is displayed.

3. Fill out the displayed fields:
 - **Name:** The name set of the card item;
 - **Insurance Type:** Choose the type of insurance from the option set. The insurance type needs to be pre-configured as per the steps detailed [here](#).
 - **Display Name:** Input the name of the card item to be displayed in the user interface, in the Quote&Bind offer cards;
 - **Code:** Assign a code for the card item in order to make it easily recognizable;
 - **Insurance Product:** The correlated insurance product;
 - **Insurance Item:** The correlated insurance item;
 - **Icon:** Upload an icon which will appear in the offer card near its item, in the Quote&Bind application;
 - **Currency:** The currency set for the insurance item according to which the premium amount is calculated taking into account the exchange rate in case of other currencies in the Quote&Bind bidding process;
 - **Description:** Free text field to describe the configuring card item;
 - **Description HTML:** Specific area for describing the card item, description which is displayed in the Quote&Bind application when clicking on the desired Offer Card. When selecting the desired card, the user is able to read the description for all the included card items.
4. Click **Save and Reload**. The **Card Item Dimension** grid is unfolded.
5. Click **Insert** to add a new card item dimension. The **Card Item Dimension** screen is displayed.

- Fill out the **Name** and **Display Name** fields for the card item dimension.

Card Item Dimension

Card Item: ExtendedInsurance

Name: Name Displayname: Display Name

Please use "Save and reload" in order to be able to add description and parameters' values

- Click **Save and Reload**. A grid is displayed, presenting the **Key** and **Value**. The keys are automatically retrieved from the Business Formulas, as every item has its own calculation formula.

Card Item Dimension

Card Item: ExtendedInsurance

Name: Name Displayname: Display Name

Description: Description

Description formula: [Formula Editor]

Key: [Dropdown Menu]

Value: [Dropdown Menu]

- Click **Save and Close**. The card item is now displayed in the **Card Items List** grid.

Card Items list

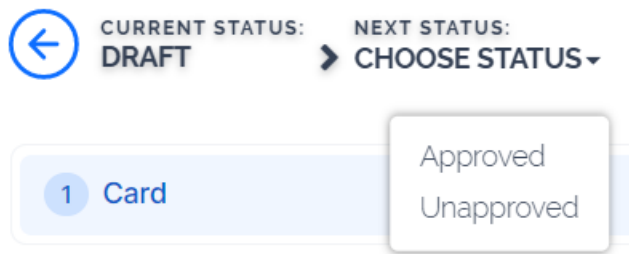
Name	Insurance Type	Displayname	Code	Insurance Product
Extended Insurance	Home	EXTENDED HOME INSURANCE	EXTH	Embedded
Home Assistance	Home	Home Assistance	HHHA	Household Insurance
Household Insurance IP	Home	Household Insurance IP	HHHA IP	Household Insurance IP
IP_test	Home	IP_test	IP_test	IP Household Insurance
J-3rdCardItem	Home	J-3rdCardItem	J-3rdCardItem	J Coverage

Configure Cards

In this section, you can configure the cards that are displayed in the interface, in an insurance Quote&Bind service extension application, after the customer answers the DNT questionnaire. These are added to specific insurance persona, which is configured in the **Q&B Configurator > Insurance Persona** section.

1. Go to **Main Menu > Product Factory > Proposal Configurator > Cards**. The **Card List** screen is displayed.
2. Click **Insert** to add a new card.
3. In the **Card** tab, fill out the following fields.
 - **Effective Date:** The date when the specific card version is available in the system. You can choose a date which is greater or equal to the current date. The date cannot be greater than the end date of the insurance product;
 - **Name:** The name set for the configured card;
 - **Insurance Type:** Choose the type of insurance from the option set. The insurance type needs to be pre-configured as per the steps detailed in the [Insurance Product Factory](#) guide;
 - **Card Type:** Drop down to choose the card type. The possible values are Single Product and Multi Product;
 - **Product:** Drop down to choose the product for the card;
 - **Display Name:** The card name to be displayed in the user interface, in the Quote&Bind offer cards;
 - **Code:** Assign a code for the card in order to make it easily recognizable;
 - **Technical Description:** Specific area for the technical description;
 - **Description:** Free text field to input a description for the created card. This is displayed in the Quote&Bind offer cards.

4. Click **Save and Reload**. The record is saved, and now the card is in **Draft** status, as shown in the top left corner.
5. After the card is configured, you can choose the next status as **Approved** or **Unapproved**.



6. To create a new version for the card, after the card is set in the **Approved** status, click the **New Version** button on the top right corner.
7. View the versioning history of the card by accessing the **History** tab.

HISTORY					
<input type="button" value="Refresh"/>		<input type="button" value="Export"/>			
<input type="checkbox"/>	Label	Name	Attribute Version Date	Attribute Version	Modified by user
<input type="checkbox"/>	Q	Q	Q	Q	Q
<input type="checkbox"/>	Approved		01/04/2022 12:36	3	
<input type="checkbox"/>	Version Closed		01/04/2022 12:36	2	
<input type="checkbox"/>	Version Closed		01/04/2022 03:00	1	

If the **Effective Date** is further than the current date, the card passes to the **Pending** status, and becomes **Approved** when reaching the **Effective Date**.

If the **Effective Date** is equal to the current date, then the version of the card automatically passes to the **Approved** status.

When first time creating a card from draft status, set the status in **Pending**, and after that, evaluate the **Effective Date**. You can edit the card in **Pending** status.

The FTOS_IP_Approve_PendingCards job is used to approve the cards based on the **Effective Date**. Read more by accessing the [Scheduled Jobs](#) page.

The status change of the old version to the closed version is only done when the new version is approved.

After you have saved the version of the card, in the **Card** section, the **Card Dimensions**, **Insurance Persona Cards** and **Card Item Config** sections are unfolded.

Card Dimensions

You can define a dimension for the card, for **Single Product** cards with **Tariff = Per Product**. You can use some default values for the inputs of the premium calculation formula. To add a card dimension, follow the steps below:

1. In the **Card Dimensions** section, click **Insert**.
2. Fill out the **Name** and **Display Name** fields.

The screenshot shows the 'Card Dimension' configuration form. At the top, there are three buttons: 'Save and close', 'Save and reload', and 'Save and new'. Below the form title, there is a 'Card' field with the value 'LifeInsurance new version'. Underneath, there are 'Name' and 'Display Name' fields, both containing the text 'Insurance Card Dimension'. A teal banner at the bottom of the form reads: 'Please use "Save and reload" in order to be able to add description and parameters' values'.

3. Click **Save and Reload**.
4. You can now fill out the **Description** and **Description HTML** fields, and they keys from the formula attached on product level are also displayed. You can manually add a **Value** for every key, and choose the **Key Type** and **Key Sub-type**, as per below.

The screenshot shows a table for configuring keys. It has four columns: 'Key', 'Value', 'Key Type', and 'Key Sub-type'. The 'Key' column lists 'frequency', 'coverage', 'sumInsured', and 'age'. The 'Value' column is empty. The 'Key Type' column has a dropdown menu open showing options: '(All)', '[none]', 'Sum Insured', and 'Excess'. The 'Key Sub-type' column has a dropdown menu open showing options: '(All)', '[none]', 'Flat', '% of Sum Insured', and '% of Loss'.

5. Click **Save and Close**.
6. The configured card dimension is now displayed in the **Card Dimension** section.

The screenshot shows the 'Card Dimension' list. At the top, there are buttons for '+ Insert', 'X Delete', 'Export', and 'Refresh'. Below the buttons, there is a table with four columns: 'Card', 'Name', 'Display Name', and 'Input Param'. The 'Card' column has a checkbox and the value 'LifeInsurance'. The 'Name' column has the value 'Insurance Card Dimension'. The 'Display Name' column has the value 'Insurance Card Dimension'. The 'Input Param' column has the value '[[{"key": "frequency", "value": null, "masterT...']

Insurance Persona Cards

In this section, you can insert the persona to be displayed on the card. Each insurance persona is correlated with different cards. The offer cards are displayed according to the insurance persona resulting from the customer's selection of options.

For each insurance persona there can be added as many cards as necessary in a desired order and having an offer card as a highlighted offer, the most suitable offer for the customer's needs.

1. In the **Insurance Persona Cards** section, click **Insert** to add a new card.
2. Fill out the following fields.
 - **Card:** The correlated card for the created insurance persona;
 - **Insurance Persona:** Insurance persona which needs to be previously configured in the system;
 - **Highlighted:** If the field is ticked, then in the Quote&Bind application when displaying the Offer Cards, one or more Cards with this setting is highlighted as the most suitable option for the customer's needs;
 - **Name:** Name for the insurance persona card;
 - **Order:** The order in which the cards are displayed in the Quote&Bind application's interface.
3. Click **Save and Close**.

Card Item Config

1. In the **Card Item Config** section, click **Insert**.
2. Fill out the following fields.
 - **Card:** By adding card items on a card, the card name is filled in the interface by default;
 - **Card Item:** The name of the card item chosen from the already created card items in order to correlate it with the card;

- **Order:** The order in which the item appears displayed on the card;
- **Active:**
 - **Active** = the card item is included as a covered item in that offer card, meaning that it is taken into consideration for premium amount calculation.
 - **Not Active** = the card item is grayed-out in the offer card, not included in the offer for premium calculations, showing the customer what the offer is like without that item.

3. Click **Save and Reload**. The **Card Item Config X Versions** section is unfolded. Each card item can have multiple versions.
4. Click **Insert** to create a new card version.
5. Fill out the following fields.
 - **CardItemConfig:** Auto-populated field with the card item name;
 - **CardItemVersion:** Option set field to choose the card item version, after the versions are previously configured;
 - **Default:** If ticked, this version appears as the default.

6. Click **Save and Close**. The created version is now listed in the **Card Item Config X Versions** section of the **Card Item Config** screen.

CARDITEMCONFIGVERSIONS

+ Insert

X Delete

Export

Refresh

<input type="checkbox"/>	CardItemConfig	CardItemVersion	Order	Default
	Q	Q	Q	(All)
	PAD - PAD	PAD	1	<input checked="" type="checkbox"/>

7. Click **Save and Close**. The created card item is displayed in the **Card Item Config** section of the **Card** screen.

CARD ITEM CONFIG

+ Insert

X Delete

Export

Refresh

<input type="checkbox"/>	Card	CardItem	Active	Order
	Q	Q	(All)	Q
	PAD	PAD	<input checked="" type="checkbox"/>	1

8. Click the **Test Calculation** tab and input the formula in the **Calculation Values** section. To learn more about configuring Business Formulas, check the [Innovation Studio](#) guide.
9. Click **Save and Close**. The card is now listed in the **Cards List** screen.

←

+ Insert

g Delete

Export

Q Advanced find

Cards list

<input type="checkbox"/>	Name	Insurance Type	Display Name	Code 1	Business Status
	Q	Q	Q	Q	Q
	LifeInsurance	Life	Life Insurance	1489EF02-00B7-4C55-A70C-...	Approved
	HealthInsurance	Health	Health Insurance	1489EF02-1435-347-876543	Draft

Configure DNT Question Answers

In this section, you can configure the answers for the DNT questions created earlier, based on which the fitting cards are displayed. The DNT Answers represents the proper options which are used for offer calculations. This means that the registered answer has an impact on the premium amount calculation displayed for each insurance offer.

1. Go to **Main Menu > Product Factory > Proposal Configurator > DNT Question Answers**.
2. Click the **Insert** button on the top right to insert a new DNT answer.
3. Fill out the following fields.
 - **Name:** Input the DNT question answer name;
 - **DNT Question:** Input the correlated DNT question for which the answer is available;
 - **Code:** Input a specific code for the DNT question answer;
 - **Description:** Input a description for the DNT answer;
 - **Technical Description:** Input a technical description for the DNT answer;
 - **Is Default:** If checked, this answer is taken as the default answer.

The screenshot shows the 'DNT Question Answer' form. At the top, there is a back arrow icon and three buttons: 'Save and close', 'Save and reload', and 'Save and new'. The form itself has a light blue header bar with the title 'DNT Question Answer'. Below the header, the form is divided into several sections:

- Name:** A text input field containing 'No-J-Home Assistance?'.
- DNT Question:** A dropdown menu showing 'J-Home Assistance?' with a downward arrow and a pencil icon. To its right is a **Code** field with the value 'false' and a dropdown arrow.
- Description:** A text input field containing 'No'.
- Technical Description:** A larger text input field containing 'Technical Description'.
- Is Default:** A checkbox that is currently unchecked.

4. Click **Save and Close**. The answer is listed in the **DNT Question Answers** list.

DNT Question Answers list

Name	Code	DNT Question
No-J-Home Assistance?	false	J-Home Assistance?
No-J-Third Party?	false	J-Third Party?
No-Lifetime	False	Lifetime
No-Online Channel	False	Online Channel
No-Third party liability	false	Third party liability
No-Time Limited	False	Time Limited
Yes-Building	true	Building
Yes-Content	true	Content
Yes-Direct Channel	True	Direct Channel
Yes-Home Assistance	true	Home Assistance

Configure the Insurance Persona

Each audience created must be correlated with an insurance persona, which is correlated with a quote.

In this section, you can configure the types of insurance persona and the cards to be displayed.

1. Go to **Product Factory > Proposal Configurator > Insurance Persona**.
2. Click **Insert** to add a new persona.
3. In the **Insurance Persona** screen, fill out the following fields:
 - **Persona Type:** Option set field. Choose the type of insurance person:
 - **Persona** - A single segment is addressed. You need to have an existing configured audience segment;
 - **Audience** - Multiple segments are addressed. You need to have an existing configured audience.
 - **Name:** The insurance persona name;
 - **Persona:** The correlated persona in case that the Persona Type is set as Persona;

- **Audience:** The correlated audience in case that the Persona Type is set as Audience;
- **Formula:** Dropdown to choose the formula to be used for the insurance persona;
- **Description:** Input a description for the created persona.

The screenshot shows the 'Insurance Persona' configuration interface. At the top, there's a title bar with a back arrow on the left and three buttons: 'Save and close', 'Save and reload', and 'Save and new'. Below the title bar, the form is organized into two columns. The left column contains 'Persona Type' (a dropdown menu currently showing 'Audience'), 'Persona' (a dropdown menu), 'Formula' (a dropdown menu), and 'Description' (a text area). The right column contains 'Name' (a text input field with 'Basic Audience') and 'Audience' (a dropdown menu currently showing 'Basic Audience'). At the bottom of the form, there is a section labeled 'Cards'.

4. Click **Save and Reload**. The **Cards** grid is unfolded.
5. In the **Cards** grid. Click **Insert** to add the cards to be displayed for the created persona. Choose the cards from the given list.
6. The **Insurance Persona Card** screen is displayed. Fill out the following fields:
 - **Card:** The correlated card for the created insurance persona;
 - **Insurance Persona:** Insurance persona configured existing in the system;
 - **Highlighted:** If the field is ticked, then in the Quote&Bind application when displaying the Offer Cards, one or more Cards with this setting will be highlighted as the best option for the customer's needs;
 - **Name:** Name for the insurance persona card;
 - **Order:** The order in which the cards are displayed in the Quote&Bind application's interface.

Save and close

Save and reload

Insurance Persona Card

Card

Card-SingleProduct

Insurance Persona

Basic HHI Audience

Highlighted

Order

1

7. Click **Save and Close**. The created insurance persona card is displayed in the **Cards** section.
8. Set the order of the cards in this section by using drag and drop. Choose which of the displayed cards is highlighted in the interface by ticking the **Highlighted** box of the specific card.

Cards

+ Insert

X Delete

Export

Refresh

Insurance Persona	Card	Highlighted	Order
<input type="text"/>	<input type="text"/>	(All)	<input type="text"/>
<input checked="" type="checkbox"/> Basic Audience	Card-SingleProduct	<input type="checkbox"/>	1
Basic Audience	Extended	<input type="checkbox"/>	2

9. Click **Save and Close**. The created persona is displayed in the **Insurance Persona List** grid.

+ Insert

Delete

Export

Advanced find

Insurance Persona list

Name	Persona Type	Persona	Audience
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input checked="" type="checkbox"/> Basic HHI Audience	Audience		Basic HHI Audience
Complete HHI Audience	Audience		Complete HHI Audience
DNT Building	Persona	DNT Building	
DNT Home Assistance	Persona	DNT Home Assistance	

Proposal Configurations

Proposal Configurator comes with a series of an API and a series of endpoints used to calculate the premiums that are displayed on the offer cards, and to generate card details.

Proposal Configurator API	32
Proposal Configurator Endpoints	68
Proposal Configurator Scheduled Jobs	85

Proposal Configurator API

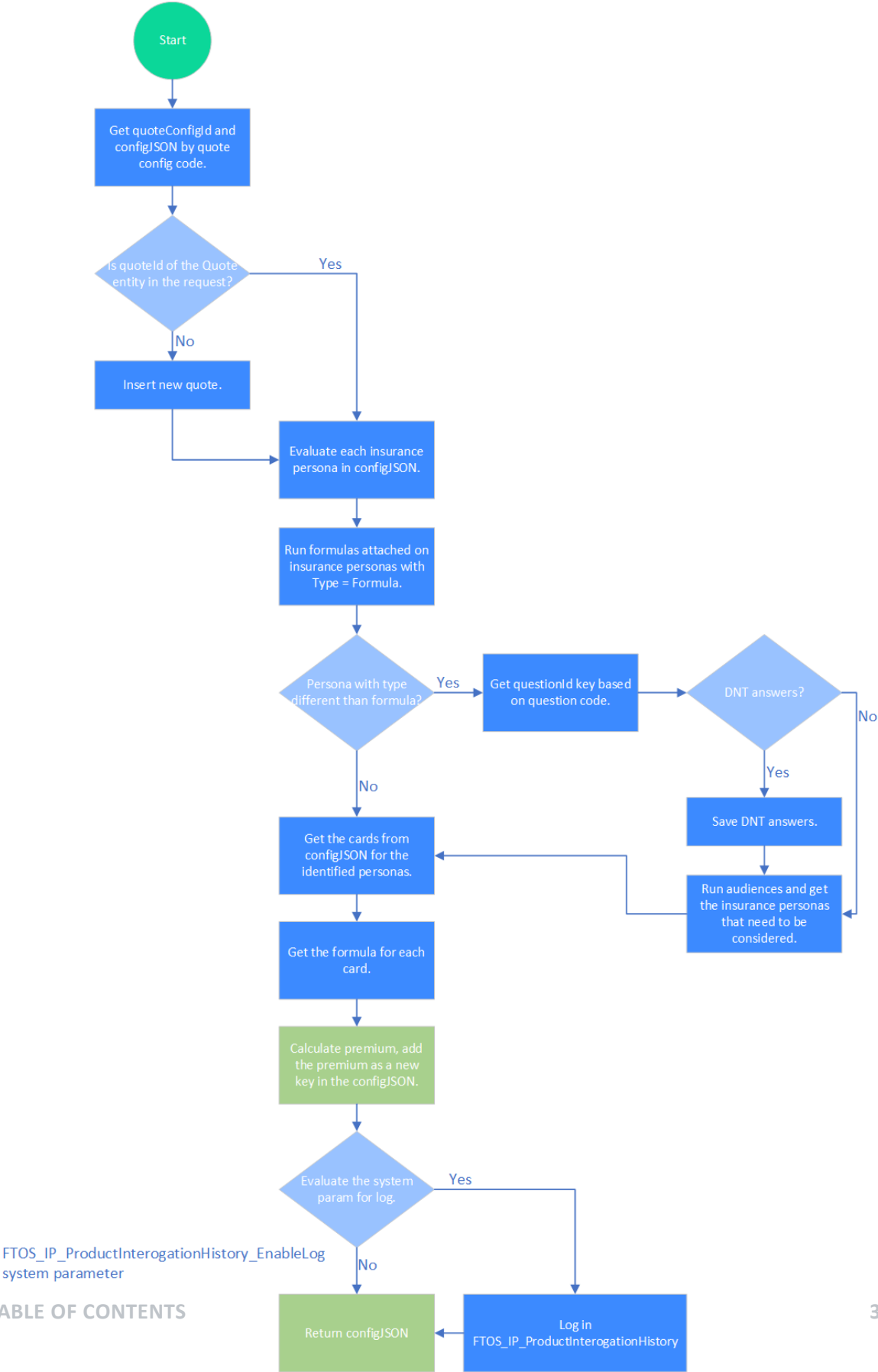
The `FTOS_IP_ProposalConfigPremiumCalculation` API was designed to allow the interaction with the **Proposal Configurator** solution, returning details about the available cards and their premiums, based on the responses provided to the DNT questions. The current versions of the API can be used in the quotation process, to get the premium amount for each card, using the latest settings created for a specific quote config.

In order to get fast responses using the API, all the quote config details are stored in a JSON object in the `configJSON` attribute of Quote Config entity. Every time a change occurs on the configs, this attribute is updated (adding/removing a new insurance persona on card config, adding/removing card from insurance personas, approving new versions of the cards, etc).

The update of the `configJSON` attribute is handled by the `updateConfigJSON` function from the `FTOS_IP_ProposalConfigPremiumCalculationAPI` server automation scripts library, described here.

All the requests send using this API and their responses can be saved into the Product Interogation History entity. This action is available if the system parameter `FTOS_IP_ProductInterogationHistory_Enablelog` value is set to 1.

The logic of the API is covered in the next diagram:



Request and Response Examples

The following examples are using the `ebs.callActionByNameAsync` Fintech SDK function.

Tariff Per Product Request

Tariff Per Product Request

```

1  let p = {
2    "quoteConfigCode": "PET",
3    "validityDate": null,
4    "quoteList" : [{
5      "quoteNumber":
6        "Quote123",
7      "quoteId": "383e5ed0-
8        3f6b-40c6-b67f-4b80322ea058",
9      "premiumCalculationDetails" : [{
10     "calculationDetails" :
11     {
12       "insuredAmount": null,
13       "breed":
14       "Cat",
15       "voluntaryExcess": 100,
16       "policyTerm": 10,
17       "petAge":
18       5,
19       "group":
20       25,
21       "product": "Lifetime5k",
22       "area":
23       17,
24       "lifetype": "Cat",
25       "breedName": "Boss",
26       "neutered": true,
27       "gender":
28       "Female",

```

```

21  "lagDays": 2,
22                                     "price":
23  122,
24  "numberOfPets": 3,
25  "mixOfPets": "Mixed",
26                                     "claims":
27  {
28    "isNewBusiness": false,
29    "nonReoccurring": 150,
30    "potentialReoccurrence": 700,
31    "reoccurrence": 1500,
32    "qtNonReoccurring": 1,
33    "qtPotentialReoccurrence": 2,
34    "qtReoccurrence": 3,
35    "isClean": true,
36    "previousPremium": 650
37                                     },
38  "isMultipet": true,
39  "salesChannel": "TEL",
40  "eligibleDiscounts": [  {
41    "discount": "Existing Customer",
42    "value": 0
43                                     },  {
44    "discount": "Staff",
45    "value": 0

```

```

44         }],
45     "postcodeArea": "IPT"
46     },
47     }
48   ]
49 },
50 "DNTResponses": [
51   {
52     questionCode: 'PET_DIR', // direct
53     answerValue: false
54   },
55   {
56     questionCode: 'PET_ONL', // online
57     answerValue: true
58   }
59   ,
60   {
61     questionCode: 'PET_LFT', // time limited
62     answerValue: true
63   }, {
64     questionCode: 'PET_LIM', // life time
65     answerValue: true
66   }
67 ]
68 ];
69
70 ebs.callActionByNameAsync("FTOS_IP_
71 ProposalConfigPremiumCalculation", p).then
72 (function(e){console.log(e)}).catch(function
73 (err) { console.log(err) });

```

Tariff Per Product Response

Tariff Per Product Response

```

1  {
2    "isSuccess":true,
3    "errorMessage":null,
4    "errorCode":null,
5    "result":[

```



```
6      {
7        "quoteNumber": "Quote123",
8        "quoteId": "383e5ed0-3f6b-40c6-b67f-
9        4b80322ea058",
10       "quoteConfigCode": "PET",
11       "mainProductCode": "PET",
12       "mainProductName": "Pet Product",
13       "mainProductAttributeVersion": 1,
14       "packages": [
15         {
16           "cardId": "d856fe97-fe2c-45e3-
17           bfd4-9a11030c3654",
18           "cardDisplayName": "Blue",
19           "cardName": "Lifetime2k",
20           "cardCode": "PET_L_2",
21           "cardTypeId": "feced3bc-934e-492f-
22           85aa-0a409df13c9f",
23           "productId": "9e1f9a36-fb6f-40de-
24           a550-3e7a69ea33f5",
25           "productTariffTypeId": "f724bd42-
26           2f6e-4cc1-b399-117132be310e",
27           "isHightlighted": null,
28           "personaCardOrder": 1,
29           "formulaName": "BluePetRating",
30           "dimensions": [
```

```

27 |                                     "versionParams":["\n
    | {\n          \\"key\\":
    | \\"numberOfPets\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"WholeNumber\\",\n          \\"objProps\\":
    | null\n    },\n    {\n          \\"key\\":
    | \\"group\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"WholeNumber\\",\n          \\"objProps\\":
    | null\n    },\n    {\n          \\"key\\":
    | \\"postcodeArea\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"Text\\",\n          \\"objProps\\": null\n    },\n
    | {\n          \\"key\\":
    | \\"area\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"WholeNumber\\",\n          \\"objProps\\":
    | null\n    },\n    {\n          \\"key\\":
    | \\"breed\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"Text\\",\n          \\"objProps\\": null\n    },\n
    | {\n          \\"key\\":
    | \\"claims\\",\n          \\"value\\":
    | null,\n          \\"masterType\\":
    | \\"SimpleType\\",\n          \\"subType\\":
    | \\"Object\\",\n          \\"objProps\\": \\"
    | {\n\n\n
    | \\"isNewBusiness\\\"\n
    | :\n
    | 3\n
    | ,\n\n\n
    | \\"nonReoccurring\\\"\n
    | :\n
    | 1\n
    | ,\n\n\n
    | \\"potentialReoccurrence\\\"\n
    | :\n
    | 1\n
    | \n
    | \\"reoccurrence\\\"\n
    | :\n
    | 1

```

```

    | \\"qtNonReoccurring\\\"

```

```

    | \\"qtPotentialReoccurrence\\\"

```

```

    | \n

```

```

    | \\"qtReoccurrence\\\"

```

```

subType\:
  \WholeNumber\","\n      \objProps\:
  null\n      },\n      {\n      \key\:
  \policyTerm\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \WholeNumber\","\n      \objProps\:
  null\n      },\n      {\n      \key\:
  \isMultipet\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \Bool\","\n      \objProps\: null\n      },\n
  {\n      \key\:
  \lifetype\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \Text\","\n      \objProps\: null\n      },\n
  {\n      \key\:
  \eligibleDiscounts\","\n      \value\:
  null,\n      \masterType\:
  \Collection\","\n      \subType\:
  \Object\","\n      \objProps\: \
  {\n      \discount\:\\":6,\n      \value\:\\":2}\n      },\n
  {\n      \key\:
  \gender\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \Text\","\n      \objProps\: null\n      },\n
  {\n      \key\:
  \breedName\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \Text\","\n      \objProps\: null\n      },\n
  {\n      \key\:
  \voluntaryExcess\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \WholeNumber\","\n      \objProps\:
  null\n      },\n      {\n      \key\:
  \price\","\n      \value\:
  null,\n      \masterType\:
  \SimpleType\","\n      \subType\:
  \
  Decimal\","\n      \objProps\>null\n      }\n]":
  key\:

```

masterType\:

WholeNumber\","\n \objProps\:

isMultipet\","\n \value\:

SimpleType\","\n \subType\:

key\":

```

28     null\n    }\n]"
29         "name":"V2000",
30         "displayName":"V2000",
31         "description":null,
32         "descriptionHTML":null
33     },
34     "insuredAmount":"2000",
35     "excess":null,
36     "excessType":null,
37     "cardItems":[
38         {
39             "cardItemId":"b70737d5-
40             6e8c-483e-8668-33f79268e916",
41             "code":"C15",
42             "displayName":"Veterinary
43             Fees",
44             "icon":"[]",
45             "descriptionHTML":null,
46             "description":null,
47             "isActive":true,
48             "cardItemOrder":1,
49             "insuranceProductId":"9e1f9a36-fb6f-40de-a550-
50             3e7a69ea33f5",
51             "insuranceItemId":"afd9f12a-db61-4844-9c80-
52             a7b99e3bf8f7",
53             "insuranceProductItemCode":"C15",
54             "premiumSplitPercentage":1.67,
55             "cardItemConfigId":"c7561b8c-08cf-4976-a446-
56             18970fffe830",
57             "subCoverages":[
58                 {
59                     "subcoverageName":"Veterinary Fees",
60                     "subcoverageCode":"SC1",
61                     "subcoverageDescription":null
62                 },

```

```

58         {
59         "subcoverageName": "CT/MRI scans and Associated
60         Costs",
61         "subcoverageCode": "SC2",
62         "subcoverageDescription": null
63         },
64         {
65         "subcoverageName": "Cruciate Ligament",
66         "subcoverageCode": "SC4",
67         "subcoverageDescription": null
68         },
69         {
70         "subcoverageName": "Dentistry as a direct result
71         of an accident",
72         "subcoverageCode": "SC3",
73         "subcoverageDescription": null
74         },
75         {
76         "subcoverageName": "Overseas Travel Extension (30
77         day limit)",
78         "subcoverageCode": "SC5",
79         "subcoverageDescription": null
80         },
81         "insuredAmount": "2000",
82         "excess": "99",
83         "excessType": "flat",
84         "dimensions": [

```

```

84 |                                     "versionParams":["\n
    | {\n          \"key\":
    | \"insuredAmount\", \n          \"value\":
    | \"2000\", \n          \"masterType\":
    | \"SimpleType\", \n          \"subType\":
    | \"Decimal\", \n          \"objProps\":
    | null, \n          \"keyType\":
    | \"sumInsured\" \n          }, \n          {\n          \"key\":
    | \"excess\", \n          \"value\":
    | \"99\", \n          \"masterType\":
    | \"SimpleType\", \n          \"subType\":
    | \"Decimal\", \n          \"objProps\":
    | null, \n          \"keyType\":
    | \"excess\", \n          \"keySubType\":
    | \"flat\" \n          } \n ]\",
85 |                                     \"isDefault\":true,
86 |                                     \"name\":\"PET-
    | Veterinary Fees-2000\",
87 |
    | \"displayName\":\"V2000\",
88 |                                     \"description\":null,
89 |
    | \"descriptionHTML\":null
90 |                                     }
91 |                                     ],
92 |                                     \"premiumAmount\":6.53
93 |                                     },
94 |                                     {
95 |                                     \"cardItemId\":\"bf522d97-
    | 74bb-441c-91ad-e081c2b400bd\",
96 |                                     \"code\":\"C18\",
97 |                                     \"displayName\":\"Co-
    | Insurance\",
98 |                                     \"icon\":null,
99 |                                     \"descriptionHTML\":null,
100 |                                    \"description\":null,
101 |                                    \"isActive\":true,
102 |                                    \"cardItemOrder\":4,
103 |
    | \"insuranceProductId\":\"9e1f9a36-fb6f-40de-a550-
    | 3e7a69ea33f5\",
104 |
    | \"insuranceItemId\":\"d7b10f03-0792-4404-8271-
    | 4f9a515761ee\",

```

```

105     "insuranceProductItemCode": "C18",
106     "premiumSplitPercentage": 8.33,
107     "cardItemConfigId": "6c976087-97d8-4ab0-8673-
108         f21583d0a61e",
109         "subCoverages": [
110             ],
111             "insuredAmount": null,
112             "excess": null,
113             "excessType": null,
114             "dimensions": [
115                 ],
116                 "premiumAmount": 32.56
117             }
118         ],
119         "premiumAmount": 39.09,
120         "formulaDetails": [
121             {
122                 "formulaResult": {
123                     "IsSuccess": true,
124                     "ErrorMessage": null,
125                     "Result": 39.09,
126                     "Input": {
127                         "insuredAmount": 2000,
128                         "breed": "Cat",
129                     }
130                 }
131             }
132         ],
133         "voluntaryExcess": 100,
134         "policyTerm": 10,
135         "petAge": 5,
136         "group": 25,
137         "product": "Lifetime5k",
138         "area": 17,
139         "lifetype": "Cat",
140         "breedName": "Boss",
141         "neutered": true,
142         "gender": "Female",
143         "lagDays": 2,
144         "price": 122,

```

```

142         "numberOfPets":3,
143         "mixOfPets":"Mixed",
144         "claims":{
145
146             "isNewBusiness":false,
147
148             "nonReoccurring":150,
149
150             "potentialReoccurrence":700,
151
152             "reoccurrence":1500,
153
154             "qtNonReoccurring":1,
155
156             "qtPotentialReoccurrence":2,
157
158             "qtReoccurrence":3,
159             "isClean":true,
160
161             "previousPremium":650
162         },
163         "isMultipet":true,
164         "salesChannel":"TEL",
165         "eligibleDiscounts":[
166             {
167                 "discount":"Existing Customer",
168                 "value":0
169             },
170             {
171                 "discount":"Staff",
172                 "value":0
173             }
174         ],
175         "postcodeArea":"IPT",
176
177         "cardName":"Lifetime2k",
178
179         "BaseRate":42.1746,
180
181         "ProductXBreed":1.4041,
182
183         "PolicyTerm":1.0394,
184
185         "VoluntaryExcessXAge":0.9669,
186
187         "Breed":0.9235,

```



```

174         "Area":1.411,
175         "AgeAtInception":1.1,
176         "AgeInMonths":60,
177
178         "AgeXSpeciesXLifetime":1.231,
179         "Neutered":0.9,
180         "UWAdjustment":1,
181         "PetPurchasePrice":1,
182         "LagDays":1.04,
183         "NumberOfPets":1,
184         "MixOfPets":1,
185
186         "RiskRate":98.28874384067849,
187         "ClaimsLoading":1.65,
188
189         "RiskRateRenewal":162.17642733711952,
190         "FixedExpenses":0,
191
192         "VariableExpenses":0.85,
193         "Commision":0.92,
194
195         "GrossPremiumPreConstraint":20.738673572521677,
196
197         "CollaringAndCapping":702,
198
199         "GrossPremiumNetIPT":702,
200
201         "MultipetDiscount":0.15,
202
203         "EligibleOtherDiscountValues":[
204             0.1,
205             0.15
206         ],
207
208         "OtherDiscounts":0.15,
209
210         "IPT":1.12,
211         "GrossPremium":56.80,
212
213         "GrossPremiumMonthly":4.7,
214
215         "GrossPremiumFinal":56.80,
216
217         "ReportValueIPT":6.81,

```

```

204 | "ReportRecalculateCommission":3.99,
205 | "ReportRecalculateExpenses":6.899,
206 |                                     "NetPremium":39.09
207 |                                     }
208 |                                 }
209 |                             }
210 |                         ]
211 |                     }
212 |                 ]
213 |             }
214 |         ]
215 |     }

```

Tariff Per Coverage Request

Tariff Per Coverage Request

```

1 | let p = {
2 |     "quoteConfigCode": "HHI",
3 |     "validityDate": null,
4 |     "mainEntityName": "FTOS_INSQB_QuoteProperty",
5 |     "mainEntityId": "9eabf832-5c37-4b9c-9cee-
6 |     70a121ec846a",
7 |     "quoteList": [{
8 |         "quoteNumber": "123ASD",
9 |         "quoteId": "741eb23b-c169-4c18-b502-
10 |        62dceb06bfdc",
11 |         "premiumCalculationDetails": [{
12 |             "itemCode": "BL01",
13 |             "calculationDetails": {
14 |                 "buildingType": "Apartment",
15 |                 "constructionYear": 2000,
16 |                 "coverage": "Building",
17 |                 "frequency": "monthly",
18 |                 "insuredAmount": 70000.0,
19 |                 "resistanceStructure":
20 |                 "Concrete",
21 |                 "usageType": "Main
22 |                 residence",
23 |                 "cardName": null
24 |             }
25 |         }
26 |     }
27 | }

```

```

21
22     },
23     {
24         "itemCode": "CN01",
25         "calculationDetails": {
26             "buildingType": "Apartment",
27             "constructionYear": 2000,
28             "coverage": "Content",
29             "frequency": "monthly",
30             "insuredAmount": 70000.0,
31             "resistanceStructure":
32                 "Concrete",
33                 "usageType": "Main
34                 residence",
35                 "cardName": null
36             }
37         },
38         {
39             "itemCode": "HA01",
40             "calculationDetails": {
41                 "buildingType": "Apartment",
42                 "constructionYear": 2000,
43                 "coverage": "Home
44                 Assistance",
45                 "frequency": "monthly",
46                 "insuredAmount": 70000.0,
47                 "resistanceStructure":
48                 "Concrete",
49                 "usageType": "Main
50                 residence",
51                 "cardName": null
52             }
53         },
54         {
55             "itemCode": "TL01",
56             "calculationDetails": {
57                 "buildingType": "Apartment",
58                 "constructionYear": 2000,
59                 "coverage": "Third party
60                 liability",
61                 "frequency": "monthly",

```

```

58         "insuredAmount": 70000.0,
59         "resistanceStructure":
60         "Concrete",
61         "usageType": "Main
62         residence",
63         "cardName": null
64     }
65 }
66 },
67 "DNTResponses": [{
68     questionCode: 'CNT',
69     answerValue: true
70 },
71 {
72     questionCode: 'HA',
73     answerValue: false
74 },
75 {
76     questionCode: 'TPL',
77     answerValue: false
78 }, {
79     questionCode: 'BLD',
80     answerValue: true
81 }
82 ]
83 ];
84
85 ebs.callActionByNameAsync("FTOS_IP_
86 ProposalConfigPremiumCalculation", p).then
87 (function(e){console.log(e)}).catch(function
88 (err) { console.log(err) });

```

Tariff Per Coverage Response

Tariff Per Coverage Response

```

1  {
2      "isSuccess":true,
3      "errorMessage":null,
4      "errorCode":null,

```

```

5      "result":[
6          {
7              "quoteNumber":"123ASD",
8              "quoteId":"741eb23b-c169-4c18-b502-
62dceb06bfdc",
9              "quoteConfigCode":"HHI",
10             "mainProductCode":"Household
Insurance",
11             "mainProductName":"Household
Insurance",
12             "mainProductAttributeVersion":3,
13             "packages":[
14                 {
15                     "cardId":"0955c095-eeac-447a-
a7c8-5151280dc5f6",
16                     "cardDisplayName":"COMPLETE",
17                     "cardName":"Complete",
18                     "cardCode":"D0955C095-EEAC-447A-
A7C8-5151280DC5F6",
19                     "cardTypeId":"37249283-b147-4001-
8ef0-55466c8cb96a",
20                     "productId":null,
21                     "productTariffTypeId":null,
22                     "isHightlighted":false,
23                     "personaCardOrder":3,
24                     "formulaName":null,
25                     "dimensions":[
26
27                     ],
28                     "insuredAmount":null,
29                     "excess":null,
30                     "excessType":null,
31                     "cardItems":[
32                         {
33                             "cardItemId":"4a4ad95f-
04f4-4f31-8d46-0d46e319cbd0",
34                             "code":"HHIBLD",
35                             "displayName":"Buildings",
36                             "icon":"[]",
37                             "descriptionHTML":null,
38                             "description":null,
39                             "isActive":true,
40                             "cardItemOrder":1,

```

```

41  "insuranceProductId":"8257c205-7c97-4563-85a6-
    ee4e0b1c3a53",
42  "insuranceItemId":"8f833fa1-652b-4985-ad3a-
    c02f00b19975",
43  "insuranceProductItemCode":"BL01",
44  "premiumSplitPercentage":null,
45  "cardItemConfigId":"b34e8df7-2b98-4d67-9dbd-
    7a34a4a7d091",
46      "subCoverages":[
47          {
48  "subcoverageName":"Theft",
49  "subcoverageCode":null,
50  "subcoverageDescription":null
51          },
52          {
53  "subcoverageName":"FLEXA",
54  "subcoverageCode":null,
55  "subcoverageDescription":null
56          },
57          {
58  "subcoverageName":"Other Natural Disasters",
59  "subcoverageCode":null,
60  "subcoverageDescription":null
61          },
62          {
63  "subcoverageName":"Malicious acts",
64  "subcoverageCode":null,

```

```

65   "subcoverageDescription":null
66       },
67       {
68   "subcoverageName":"Water Damage",
69   "subcoverageCode":null,
70   "subcoverageDescription":null
71       },
72       {
73   "subcoverageName":"Nat-Cat",
74   "subcoverageCode":null,
75   "subcoverageDescription":null
76       }
77   ],
78   "insuredAmount":null,
79   "excess":null,
80   "excessType":null,
81   "dimensions":[
82   ],
83   "premiumAmount":90.55
84   },
85   ],
86   "premiumAmount":90.55,
87   "formulaDetails":[
88   {
89       "itemCode":"BL01",
90       "formulaResult":{
91           "IsSuccess":true,
92           "ErrorMessage":null,
93           "Result":90.552,
94           "Input":{
95   "buildingType":"Apartment",
96   "constructionYear":2000,
97   "coverage":"Building",

```

```

99      "frequency": "monthly",
100     "insuredAmount": 70000,
101     "resistanceStructure": "Concrete",
102                                "usageType": "Main
residence",
103                                "cardName": null,
104     "BuidingSumInsured": 1.1,
105                                "BuildingType": 0.7,
106     "ResistanceStructure": 0.7,
107                                "ConstructionYear": 1,
108                                "UsageType": 1,
109                                "Frequency": 1.2,
110                                "FinalCoef": 0.6468,
111                                "BaseRate": 0.002,
112     "FinalRate": 0.0012936,
113     "PremiumAmount": 90.552
114   }
115 }
116 }
117 ]
118 }
119 ]
120 }
121 ]
122 }

```

Request Data Parameters

Parameter	Description
quoteConfigCode	Code of the quote config used to identify the settings from the Proposal Configurator module.

Parameter	Description
validityDate	<p>The reference date prior to the current date, used if you want to run the premium calculation formulas at a specific moment of time. The value is passed only to the formula.</p> <p>Use null as a value to use the current approved version of the formula.</p> <p>Format: YYYY-MM-DD</p>
mainEntityId	<p>Unique identifier or a record or null. Key used only if the proposal configuration includes insurance personas of type Persona or Audience.</p> <p>Record ID of the target entity instance that you wish to check whether it is associated with the audience/insurance persona.</p>
DNTResponses:	<p>Array of objects with details about the answers to DNT questions. Check DNTResponse object structure.</p> <p>For insurance personas of types audience/persona:</p> <ul style="list-style-type: none"> • if the array contains objects, than they are used to save the answers in the DNT Question Provided Answer entity; • if the answers are already saved in the DNT Question Provided Answer entity from your flow, than you can send []. <p>For insurance personas of type formula:</p> <ul style="list-style-type: none"> • the objects from the array are used to determine the input of the formula attached on the insurance persona, each object of the array will become a input key for the formula: <ul style="list-style-type: none"> • questionCode becomes the key in the formula input; • objectanswerValue becomes the value in the formula input object .

Parameter	Description
quoteList	<p>Array of objects containing details about individual quotes. Check Quote object structure. For simple quotes the array contains only one element.</p> <p>For master quotes, details about each individual quotes can be sent at once, if the DNT questions are the same for each of the quotes included in this array (we will evaluate the insurance personas only once, and the identified insurance personas will be used for all the objects included in this array).</p>

DNTResponse Object Structure

Parameter	Description
questionCode	Code of the DNT question included in the quote config identified by quoteConfigCode.
answerValue	Answer to the question identified by questionCode, it can be either true or false.

Quote Object Structure

Parameter	Description
quoteNumber	Number of the quote as allocated in the quotation process.
quoteId	<p>Optional parameter, the uniqueidentified of a record from the Quote entity or null.</p> <p>If this key is null, then a new record is added in the Quote entity, saving just the name of the quote; the DNT answers are saved for this new record.</p> <p>If the Proposal Configurator solution and the quote flow are installed on the same instance, than a record already exists in the Quote entity, from the quotation process; pass it as a parameter.</p>

Parameter	Description
premiumCalculationDetails:	Array of objects used to run the premium calculation formulas.
itemCode	Mandatory only for products with Tariff = Per Coverage . Item code - the key is not available for products that have Tariff Type = Per Product .
calculationDetails	Object containing the keys needed to run the formula. The object structure is the one used to test the formula, it can be different for each item, based on the formula configuration

Response

Response Description

Key	Description
Error code	Error code.
Error message	Error message.
isSuccess	Marks if the request was successful or not.
result	Array of objects containing details about the prices, for each of the objects included in the quoteList array from the request.
quoteNumber	Number of the quote as allocated in the quotation process.
quoteId	ID of the quote. Either the quoteId passed as a parameter in the request of the new id of the records automatically inserted.
quoteConfigCode	Quote Config Code used to by the API. It has the same value as quoteConfigCode key from the request object.
mainProductCode	The code of the product selected as main product on quote config.

Key	Description
mainProductName	The name of the product selected as main product on quote config.
mainProductAttributeVersion	The version of the product selected as main product on quote config.
packages	Array of objects describing each of the available packages (cards). Check Package (card) object structure.

Package (card) object structure

Key	Description
cardId	Unique identifier of the record from the Card entity.
cardDisplayName	Display name attribute value from the Card entity.
cardName	Name attribute value from the Card entity.
cardCode	Code attribute value from the Card entity.
cardTypeId	Card type attribute value from the Card entity. Unique identified representing the ID of the selected option set item.
productId	Product attribute value from the Card entity. Unique identified representing the ID of the selected product.
productTariffTypeId	Type of tariff from the product selected in the Product attribute of the card. Unique identified representing the ID of the selected option set value from tariff type attribute on product level.
formulaName	For products with tariff type = per product, name of the formula linked to the product. For products with tariff type = per coverage, null.
isHighlighted	Highlighted attribute value from the Insurance Persona Card entity record, associated with the card.
personaCardOrder	Attribute value from the Insurance Persona Card entity record, associated with the card.
dimensions	Array of objects describing the dimensions set on the card. Check Dimension object structure.

Key	Description
insuredAmount	Value of the card dimension key marked as "Sum Insured".
excess	Value of the card dimension key marked as "Excess".
excessType	Subtype of the card dimension marked as "Excess".
cardItems	Array of objects describing the card items included in the card. Check Card item object structure.
premiumAmount	Premium amount of the card.
formulaDetails	<p>Array of objects describing the full results of the formula, useful for debugging purposes.</p> <p>Contains only one element for tariff = per product and multiple elements for tariff = per coverage - one element for each item.</p>

Card Item Object Structure

Key	Description
cardItemId	Unique identifier of the record from the Card Item entity.
code	Code attribute value from the Card Item entity.
displayName	Display name attribute value from the Card Item entity.
icon	Icon attribute value from FTOS_IP_CardItem entity.
descriptionHTML	Description HTML attribute value from FTOS_IP_CardItem entity.
description	Description attribute value from FTOS_IP_CardItem entity.
insuranceProductId	<p>Insurance Product attribute value from FTOS_IP_CardItem entity.</p> <p>Uniqueidentified representing the id of the selected product.</p>
insuranceItemId	<p>Insurance Item attribute value from FTOS_IP_CardItem entity.</p> <p>Uniqueidentified representing the id of the selected insurance product item.</p>
insuranceProductItemCode	Code of the selected insurance item.

Key	Description
cardItemConfigId	Unique identifier of the record from the Card Item Config, representing the link between the card and the card item.
premiumSplitPercentage	Percentage attribute value from the Card Item Config entity.
isActive	Active attribute value from the Card Item Config entity.
cardItemOrder	Order attribute value from the Card Item Config entity.
dimensions	Array of objects describing the dimensions set on the card item. Check Dimension object structure.
insuredAmount	Value of the dimension key marked as “Sum Insured”, identified from the dimension object marked as default.
excess	Value of the dimension key marked as “Excess”, identified from the dimension marked as default.
excessType	Subtype of the dimension marked as “Excess”, identified from the dimension object marked as default.
premiumAmount	Premium amount of the card item.
subCoverages	Array of objects describing a sub coverage (module) of the product coverage selected on the card item.

Sub Coverage Object Structure

Key	Description
subcoverageName	Name attribute of the product sub coverage (module).
subcoverageCode	Code attribute of the product sub coverage (module).
subcoverageDescription	Description attribute of the product sub coverage (module).

Dimension Object Structure

Key	Description
versionParams	The inputParams attribute of the Card Dimension or Card Item Version entity.
isDefault	Filled in for card item dimensions. Default attribute value of the Card Item Config x Version entity.
name	Name attribute value of the Card Dimension or Card Item Version entity.
displayName	Display name attribute value of the Card Dimension or Card Item Version entity.
description	Description attribute value of the Card Dimension or Card Item Version entity.
descriptionHTML	Description HTML attribute value of the Card Dimension or Card Item Version entity.

Error Messages

Code	Text	Description
ERR.PC.50101	quoteNumber is mandatory for premium calculation!	quoteNumber input key missing or empty.
ERR.PC.50102	quoteId must be uniqueidentifier type!	quoteId included in the request, but is not a uniqueidentifier.
ERR.PC.50103	quoteConfigCode is mandatory for premium calculation!	quoteConfigCode input key missing or empty.

Endpoint Technical Description

The `FTOS_IP_ProposalConfigPremiumCalculation` endpoint is used for calculating the premium amount, using the Proposal Configurator settings. The main logic of the on demand script is included in the `proposalPremiumCalculation` function of the `FTOS_IP_ProposalConfigPremiumCalculationAPI` server automation script library.

Server Side Script Libraries

`FTOS_IP_ProposalConfigPremiumCalculationAPI`

This library contains functions used by the proposal configurator API or functions used to support the logic.

The premium calculation API calls only one main function, `proposalPremiumCalculation`. The flow logic is included in it. All the other functions from this library are helpers created to run small pieces of the process, that together cover all the logic.

`proposalPremiumCalculation(context)`: This is the main function used to start the logic of the proposal configurator premium calculation API. It starts by validating the request (by calling the `validateRequest` function) and then preparing pre-requisites variables and starting the cards identification and premium calculation process (by calling the `runPremiumCalculationLogic` function).

Input parameters:

- context - context object used by the on demand server automation script.

Output parameters:

- object containing keys to describe the result of the API:
 - isSuccess - true/false;
 - errorMessage - null or error message as described in the error messages list;
 - errorCode - null or error code as describes in the error messages list;
 - result = [] or array with details about the price, as described in the **Response Description** section.

`validateRequest(context)`: This function is used to validate request fields.

Input parameters:

- context - context object used by the on demand server automation script.

Output parameters:

- object containing keys to describe the result of the validation:
 - isSuccess - true/false;
 - errorMessage - null or error message as described in the error messages list;
 - errorCode - null or error code as describes in the error messages list;
 - result = [] or array with details about the price, as described in the Response Description section.

`runPremiumCalculationLogic (context)`: This function is used for identifying the main variables used in the process and to prepare additional objects needed for the main logic to run. This function uses the `calculatePremiumForQuotes` function to get the premium calculation results for all the cards and return them.

Input parameters:

- context - context object used by the on demand server automation script.

Output parameters:

- result = [] or array with details about the price, as described in the Response Description section.

`calculatePremiumForQuotes(quoteConfigDetails, quoteList, dntResponses, formulaPersonasInput, validityDate, mainEntityName, mainEntityId)`: This unction covers the first steps of the API logic - identifying which are cards for which the premium calculation process needs to run,

based on the identified insurance personas, and then calls `calculatePremiumForCards` to calculate the premium amount for each card.

Considering the API flow diagram, this function includes:

- Evaluating the `quoteId` key value and decide if a new Quote record needs to be inserted;
- Identifying the formula or audience/persona insurance personas;
- Saving DNT answers for non-formula insurance persona;
- Identifying the cards allocated to the identified insurance personas;
- Starting the premium calculation process per card.

Input parameters:

- `quoteConfigDetails` - object containing details about the quote config used for this flow, identified by the `quoteConfigCode` request parameter, using the `getQuoteConfigDetails` function;
- `quoteList` - `quoteList` array included in the request;
- `dntResponses` - `DNTResponses` array included in the request;
- `formulaPersonasInput`- a new object created based on the `DNTResponses` values, used for running the formulas attached on formula insurance personas;
- `validityDate` - validity date input parameter;
- `mainEntityName` - obsolete parameter, null;
- `mainEntityId` - ID of the record to be evaluated, for personas/audiences insurance personas; it's the value of `mainEntityId` input parameter.

`formulaPersonasInput` additional details:

For each element, the key = the value of `questionCode` key from `DNTResponses` and the value = the value of `answerValue` key from `DNTResponses`.

Considering this `DNTResponses` array

```

1  "DNTResponses": [
2    {
3      questionCode: 'PET_DIR',
4      answerValue: false
5    }, {
6      questionCode: 'PET_ONL',
7      answerValue: true
8    } , {
9      questionCode: 'PET_LFT',
10     answerValue: true
11   }, {
12     questionCode: 'PET_LIM',
13     answerValue: true
14   }
15 ]

```

We will get this `formulaPersonasInput` object

```

1  {
2    PET_DIR: false,
3    PET_ONL: true,
4    PET_LFT: true,
5    PET_LIM: true
6  }

```

Output parameters:

- `result = []` or array with details about the price, as described in the Response Description section.

`calculatePremiumForCards(quoteId, cardsForPersona, q.premiumCalculationDetails, validityDate)`: This function is used to determine, for each card, what kind of premium calculation logic to run, based on the type of tariff, per product or per coverage, using 2 separate functions: `calculatePerProductPremium` and `calculatePerCoveragePremium`.

Input parameters:

- `quoteId` - id of the quote, passed as a parameter to the request or inserted;
- `cardsList` - list of cards identified in the `calculatePremiumForQuotes` function;
- `premiumCalculationDetails` - `premiumCalculationDetails` object from the request;
- `validityDate` - validity date input parameter.

Output parameters:

- `packages` array from the result object , as described in the Response Description section.

`calculatePerProductPremium(card, premiumCalculationDetails, validityDate)`: This function is used to calculate the premium considering a per product scenario.

It loads the card dimensions and override the `premiumCalculationDetails` keys values with the card dimensions key values (`useDimensionsAsFormulaInput` function). If a card dimension key was set to a specific value, than that value needs to be used by formula engine. It will call the formula engine to get the results for the card (`calculateCardPremium` function) and then split the premium to get premiums for each card item (`splitPremiumPerItems` function).

Input parameters:

- card - card object for which we are calculating the premium amount;
- premiumCalculationDetails - premiumCalculationDetails object from the request;
- validityDate - validity date input parameter.

Output parameters:

- N/A

`useDimensionsAsFormulaInput(dimensions, formulaInput, isCardDimension)`: This function is used to override the premiumCalculationDetails keys with the dimension keys. It is used for both card and card item processes.

Input parameters:

- dimensions - dimensions set on card item/card level;
- formulaInput - premiumCalculationDetails passed as input for premium calculation;
- isCardDimension - key used to know if the function runs for a card or for a card item.

Output parameters:

- formulaInput - object with the same structure as formulaInput object, but with different values, if dimensions existed.

`calculateCardPremium(formulaName, input, validityDate)`: This function is used to run the formula defined in the formula engine.

Input parameters:

- `formulaName` - name of the formula used for card premium calculation;
- `input`- values used for running the formula;
- `validityDate` - validity date input parameter.

Output parameters:

- `formulaResult` - calculation results as returned by the `server.formulas.runFormula` server side SDK function.

`splitPremiumPerItems(cardItemsArr, premiumAmount):`

This function is used to split the premium per items.

Input parameters:

- `cardItemsArr` - list of items defined for the card;
- `premiumAmount` - premium amount on card level.

Output parameters:

- `cardItemsArr` - card items list, containing an extra `premiumAmount` keys for each item.

`calculatePerCoveragePremium(card, premiumCalculationDetails, validityDate):` This function is used to calculate the premium considering a per coverage scenario.

For each of the card items included in the request, in the `premiumCalculationDetails` array, we override the `premiumCalculation` values with the values configured on card item dimensions (`useDimensionsAsFormulaInput` function). If a card item dimension key was set to a specific value, than that value needs to be used by formula engine. The premium calculation

formula for each coverage will run and return the results on card item level. Summing up the premium of all the card items we get the premium amount for the card.

Input parameters:

- card - card object for which we are calculating the premium amount;
- premiumCalculationDetails - premiumCalculationDetails object from the request;
- validityDate - validity date input parameter.

Output parameters:

- card -card object containing the premium amount keys.

`getQuoteConfigDetails(quoteConfigCode)`: This function is used to get details about the quote config.

Input parameters:

- quoteConfigCode - input parameter from the request.

Output parameters

- object containing details about the quote config:
quoteConfigId , configJSON, quoteConfigCode,
mainProductCode, mainProductName,
mainProductAttributeVersion.

Proposal Configurator Endpoints

The list of endpoints used by the demo Quote Property flow, developed to demonstrate how to interact with the server side libraries included in the Proposal Configurator digital asset.

Endpoints

FTOS_INSQB_GetDNTAvailableQuestions: This endpoint is used to get the DNT questions based on the `insuranceTypeId`, `quoteConfigId`, `quoteId`. It calls the `getAvailableQuestions` function from the `DNTManagementUtils_New` server side library to identify the questions and then uses the `getQuestionInformation`, `getQuestionAnswers` and `getDefaultOrSelectedQuestionAnswers` functions from the same library to prepare the final result.

It returns an array of objects containing the following keys (for each question):

- `questionInfo` - Object return by `getQuestionInformation`;
- `quoteConfigId` - ID of the quote config;
- `answers` - Array returned by `getQuestionAnswers`;
- `initialResponse` - Object returned by `getDefaultOrSelectedQuestionAnswers`.

FTOS_INSQB_SaveDNTAnswers: This endpoint is used to save the answers provided to the DNT question. If it receives the `quoteId` as an input parameter and uses the `saveDNTAnswers` function from the `DNTManagementUtils_New` server side library.

FTOS_INSQB_GetConfiguredCards: This endpoint is used to get the cards identified by the provided answers. It receives the ID of a quote property and calls the **getCardsInformationForDisplay** and **getCardItemsByCardId** functions from FTOS_IP_PersonaManagement.

FTOS_INSQB_SelectFinalCard: This endpoint is used to mark the quote version related to the selected card as the final quote (**isFinalQuote = true**).

FTOS_IP_CardItem_GetDetails: This endpoint is used to get the details of the formula attached on product level, if the tariff type is per product, or on product item level, if the tariff type is per coverage.

FTOS_IP_CardItem_CheckTariffType: This endpoint is used to get the tariff type, per coverage or per product, for a product.

FTOS_IP_UpdateDetailsJSONOnQuoteConfigs: This endpoint is used to update the configJSON object for all quote config records.

FTOS_IP_Card_CheckIsEditableStatus: This endpoint verifies if the card business status is editable or not.

FTOS_IP_Card_GetDetails: This endpoint gets the card details: Card Name, BusinessStatusId, Card Type, ProductId.

FTOS_IP_CardItem_GetInfo: This endpoint gets the card item details.

FTOS_IP_ApprovePendingCards: This endpoint approves all the pending cards having the effective date less or equal than the current date.

Server Side Script Libraries

The following server side script libraries are used in Proposal Configurator.

FTOS_IP_ProposalConfigurator

This library contains functions used to check the configuration added is correct. The following functions are available.

Class - Proposal Configurator:

`checkExistingCardInsurancePersonaCard`
`(insurancePersonaCardId, insurancePersonaId, cardId)`: This function checks if a card is already linked to an insurance persona, when adding or updating a card on an insurance persona (insures uniqueness on the combination `insurancePersonaId` and `cardId`).

Input parameters:

- `insurancePersonaCardId` - ID of the current record being added/updated;
- `insurancePersonaId` - ID of the insurance persona for which we are searching the allocations;
- `cardId` - ID of the card we are searching for.

Output parameters:

- N/A

`setNewHighlightedInsurancePersonaCard`
`(insurancePersonaCardId, insurancePersonaId)`: This function marks all the cards assigned on an insurance persona and not being highlighted, with the exception of the current `insurancePersonaCardId`, sent as a parameter.

Input parameters:

- insurancePersonaCardId - ID of an insurance persona card record that is the new highlighted one; this record will not be updated;
- insurancePersonald - ID of the insurance persona for which we are searching cards allocations.

Output parameters:

- N/A

`checkExistingVersionCardItemConfigXVersion(cardItemConfigXVersionid, cardItemConfigId, versionId)`: This function checks if a card version is already linked to a card item config, when adding or updating a version. It insures uniqueness on the cardItemConfigId and versionId combination.

Input parameters:

- cardItemConfigXVersionid - ID of the current record being added/updated;
- cardItemConfigId - ID of the card item config for which we are searching the allocations;
- versionId - ID of the version of the card we are searching for.

Output parameters:

- N/A

`setNewDefaultCardItemConfigVersion(cardItemConfigXVersionid, cardItemConfigId)`: This function marks all card versions allocated on a certain

card item config as not being default, with the exception of the current cardItemConfigXVersionid, sent as a parameter.

Input parameters:

- cardItemConfigXVersionid - ID of a card item configuration x version record that is the new default one; this record is not updated;
- cardItemConfigId - ID of the card item config for which we are searching allocations.

Output parameters:

- N/A

`checkStatusIsEditable(businessStatusId)`: This function checks if a businessStatusId given as an input is editable.

Input parameters:

- businessStatusId - ID of the business status.

Output parameters:

- N/A

`getCardDetails(cardId)`: This function returns the card details (Card Name, BusinessStatusId, Card Type, ProductId).

Input parameters:

- cardId - ID of the card.

Output parameters:

- Query results.

`getCardLastApprovedVersion`

`(masterAttributeId)`: This function returns the last approved version.

Input parameters:

- `masterAttributeId` - ID of the master attribute

Output parameters:

- Query result.

`approvePendingCards(cardId)`: This function changes the business status to **Approved** for the pending cards having the effective date less or equal to the current date.

Input parameters:

- `cardId` - ID of the card.

Output parameters:

- N/A

`getProductDetails(productId)`: This function returns the product details.

Input parameters:

- `productId` - ID of the product.

Output parameters:

- Query results.

effectiveDateCurrentDateValidation(effectiveDate): This function throws an error if the effective date is less than the current date.

Input parameters:

- effectiveDate - the date when the new version created can be active (available in the system).

Output parameters:

- N/A

effectiveDateIpEndDateValidation(effectiveDate, insuranceProductEndDate): This function throws an error if the effective date is greater than the insurance product end date.

Input parameters:

- effectiveDate - the date when the new version created can be active (available in the system);
- insuranceProductEndDate - insurance product end date.

Output parameters:

- N/A

getInsuranceProductFormulaDetailsByCardId(cardId): This function returns the Insurance Product Formula details.

Input parameters:

- cardId - ID of the card.

Output parameters:

- Query results.

Class - CardItems

getProductTariffTypeByCardItem(cardItemId):

This function returns a list with only one tariff type that was found for a product with a specific ID associated to a card item ID.

Input parameters:

- cardItemId - ID of the card item.

Output parameters:

- List with the product tariff type.

getProductTariffType(productId): This function returns a list with only one tariff type that was found for a product with a specific ID.

Input parameters:

- productId - ID of the insurance product.

Output parameters:

- List with the product tariff type.

`validateProductTariffTypeByCardItem`

`(cardItemId)`: This function returns an object that will contain the tariff type set on a specific insurance product associated to a card item ID.

Input parameters:

- `cardItemId` - ID of the card item.

Output parameters:

- `validationObj` - object containing the tariff type.

`validateProductTariffType(productId)`: This function returns an object that contains the tariff type set on a specific insurance product.

Input parameters:

- `productId` - ID of the insurance product for which the tariff type will be returned.

Output parameters:

- `validationObj` - object containing the tariff type.

`getInsuranceProductItemFormulaDetails`

`(cardItemId)`: This function returns the information about the formula attached on the product item level.

Input parameters:

- `cardItemId` - ID of the card item.

Output parameters:

- Query results.

`getInsuranceProductFormulaDetails()`: This function returns the information set on the card item input params processor from the Proposal Configurator flow settings.

Input parameters:

- N/A

Output parameters:

- Array with objects defined on the card item input params processor.

`getCardItemDetails(cardItemId)`: This function returns the card item details.

Input parameters:

- cardItemId - ID of the card item.

Output parameters:

- Query results.

FTOS_IP_PersonaManagement

Library containing functions used to read data from the Proposal Configurator entities. These functions are listed below.

`getMatchedInsurancePersona(mainEntityName, mainEntityId)`: This function is used to identify the insurance persona based on the provided answers. It identifies all the insurance personas included in the quote config selected in the

quote. For each insurance persona, the function checks if the `mainEntityId` record is in the audience or in the segment attached to the insurance persona.

Input parameters:

- `mainEntityName` - name of the entity extending the Quote entity.
- `mainEntityId` - id of a record from `mainEntityName`.

Output parameters:

- `matchedInsurancePersonald` - unique identifier, ID of the insurance persona identified or null.

`getCardsByInsurancePersonaId(insurancePersonaId):`

This function is used to get all the cards configured for a specific insurance persona.

Input parameters:

- `insurancePersonald` - ID of the insurance persona to search for.

Output parameters:

- `cardArray` - array of objects containing details about the cards, with the following keys: `cardId`, `displayname`, `code`, `isHighlighted`, `personaCardOrder`.

`insertVersionRelatedRecords(mainEntityName, mainEntityId):` This function calls the previously mentioned functions in order to calculate the premium amount for each card:

- evaluates the responses provided to the DNT questions and determines the insurance persona

- reads all the cards defined for an insurance persona and saves the details for each in the FTOS_INSQB_QuoteVersion entity, without having the premium amount calculated yet
- gets the card items for each card and calculates the premium for each card item, saving the results in FTOS_INSQB_QuoteVersionItem entity
- updates the total premium on each quote version record , previously inserted

Input parameters:

- mainEntityName - name of the entity extending the Quote entity.
- mainEntityId - ID of a record from mainEntityName.

Output parameters:

- N/A

`getCardsInformationForDisplay(mainEntityName, mainEntityId, customConditionList)`: This function returns details about the cards, including premium details saved in the Quote Version entity and information about all the premiums calculated for each card version.

Input parameters:

- mainEntityName - name of the entity extending the Quote entity;
- mainEntityId - ID of a record from mainEntityName;
- customConditionList - additional conditions used to filter the details.

Output parameters:

- cardDisplayInformation - array of objects containing the following keys:

```

1  [{
2    "quoteVersionId": "18363122-4670-44a8-
3    a04b-3a4f5e9d77f1",
4    "code": "D",
5    "displayName": "COMPLETE",
6    "premiumAmount": 134.66,
7    "currencyName": null,
8    "isHighlighted": false,
9    "orderIndex": 3.0,
10   "cardId": "0955c095-eeac-447a-a7c8-
11   5151280dc5f6",
12   "isFinalQuote": true,
13   "cardItemDisplayInformation":
14   [CardItemVersion1, CardItemVersion2]
15 }]
```

where CardItemVersion =

```

1  {
2    "quoteVersionItemId": "00564575-
3    64c9-4ebc-ae fd-3a81b9f32f37",
4    "insuranceOptionList": [
5      {
6        "insuranceOptionName":
7        "option1",
8        "sumInsured": null,
9        "premiumAmount": 115.2,
10       "isSelected": true,
11       "orderIndex": 1.0
12     },
13     {
14       "insuranceOptionName":
15       "option3",
16       "sumInsured": null,
17       "premiumAmount": 384.0,
18       "isSelected": false,
19       "orderIndex": 3.0
20     }
21   ]
22 }
```

```

18         {
19             "insuranceOptionName":
20             "option2",
21             "sumInsured": null,
22             "premiumAmount": 288.0,
23             "isSelected": false,
24             "orderIndex": 2.0
25         }
26     ],
27     "displayName": "Content",
28     "isActive": true,
29     "icon": null,
30     "descriptionHTML": null,
31     "description": null,
32     "orderIndex": 2.0
33 }

```

getCardItemsByCardId(cardId): This function is used to get all the card items (all the coverages) from a card.

Input parameters:

- cardId - id of the card for which we are searching for items.

Output parameters:

- cardItemArray - array of objects describing a card item, with the following keys: cardItemId, code, displayName, icon, descriptionHTML, description, isActive, cardItemOrder, insuranceProductId, insuranceItemId.

DNTManagementUtils_New

This library contains a set of functions used by the quote flow to interact with the Proposal Configurator settings. The functions are described below.

`getAvailableQuestions(insuranceTypeId, quoteConfigId, quoteId)`: This function returns all the DNT questions included in a quote config.

Input parameters:

- insuranceTypeId – ID of an insurance type used when searching for questions;
- quoteConfigId – ID of a quote config used when searching for questions;
- quoteId – ID of a Quote record from where the search started.

Output parameters:

- questions - array of elements containing the following keys: questionId, configCode, quoteConfigId.

`getQuestionInformation(questionId)`: This function used to get details about a DNT question.

Input parameters:

- questionId – ID of a DNT question.

Output parameters:

- questionInfo - object containing the following keys: questionId, name, description, tooltip.

`getQuestionAnswers(questionId)`: This function is used to get the available answers for a DNT question.

Input parameters:

- questionId - ID of a DNT question.

Output parameters:

- questionInfo - array of object containing the following keys: answerId, name, code, description, isDefault.

`getDefaultOrSelectedQuestionAnswers(questionId, quoteId, answerList)`: This function is used to identify is a certain answer is default of if an answer was already provided for a question.

Input parameters:

- questionId - id of a DNT question
- quoteId - id of the quote from where the search started;
- answerList - list of available answers for a DNT question, as returned by `getQuestionAnswers`; if this parameter is not provided, the function calls `getQuestionAnswers` to identify the list of answers.

Output parameters:

- questionInfo- array of object containing the following keys: answerId, name, code, description, isDefault;
- choosedAnswer - object containing the following keys :
answerId - the answer that will be automatically displayed,
isDefault - if the answer is the default one, than it cannot be changed.

`saveDNTAnswers(quoteId, dntResponses)`: This function is used to save the answers to the DNT question in the Question Provided Answer entity. Before saving the answers, it also deletes any other previously provided answers.

Input parameters:

- `quoteId` - if of the quote for which we are saving the responses;
- `dntResponses` - array of object containing the `questionId` and `answerValue` keys; answer value can be either true or false.

Output parameters:

- N/A

UtilityFunctionsLib

This library contains general functions used to check if the values are null, to validate a certain format of a date, and to validate different option set item names. The functions are described below.

FTOS_IP_ProposalConfigPremiumCalculationAPI

`getQuoteConfigByCard(cardId)`: This function returns data from the Quote Config entity using the ID of the card.

Input parameters:

- `cardId` - the ID of the card.

Output parameters:

- `query` - Array that contains an object with the following results:
 - `FTOSIPQuoteConfigId`.

`updateConfigJSON(quote)`: This function updates the JSON object from the Quote Config entity, using the ID of the quote config.

Input parameters:

- `quote` - the ID of the quote config.

Output parameters:

- N/A

Proposal Configurator Scheduled Jobs

The following scheduled jobs are used with the **Proposal Configurator** solution.

Job Name	FTOS_IP_UpdateDetailsJSONOnQuoteConfigs
Scheduled	At 6:00 AM, daily run
Description	The job runs to update the configJSON object for all records of the Quote Config entity
Schedule Services	FTOS_IP_UpdateDetailsJSONOnQuoteConfigs (on demand server automation script)

Job Name	FTOS_IP_ApprovePendingCards
Scheduled	daily run
Description	The job runs to approve all the pending cards having the effective date less or equal than the current date.
Schedule Services	FTOS_IP_ApprovePendingCards (on demand server automation script)