Insurance Product Factory 4.3.0

User Guide

TOC

Overview

Insurance Product Factory is an end-to-end solution for growing and managing your insurance products portfolio. You can create new products, modify some of the conditions for your products, or retire products. As the portfolio evolves, the solution enables you to keep your product portfolio disciplined, accessible, and comprehensible, while solidly handling the life cycle for all your different products, in an efficient manner.

Creating Products

With this **FintechOS** solution for insurance, you can evolve a diversified collection of re-usable insurance components (such as types, risks, or product items) that you can further utilize as **building blocks** for the assemblage of highly personalized insurance products. A product item can be configured to represent a specific insurance coverage, with its own accounting, calculation, and charging structure. No coding skills required! You also author and manage insurance products from **a single comprehensive digital solution** and this greatly reduces the scope for misinterpretation of the product design specifications errors - as it is still often experienced in insurance companies that use a “product coding from spec by IT” approach.

With the **Insurance Product Factory**, you can create insurance products by configuring insurance types, product items, risks, premiums, fees, conditions; and also by specifying the coverage, grace period, conditions for suspension, for generating invoices, limit the availability of your products and more. This solution integrates smoothly with **FintechOS** Business Formulas to help you reduce completion time for premium amount calculations, insurance risk rules and underwriting rules configurations or for testing different product offers. Once activated, embed your product into digital journeys to expose it to your potential customers - through different digital touchpoints, channels, or portals.

The solution also allows you, in conjunction with other **FintechOS** capabilities, to replicate your existing insurance products managed through legacy systems, into the **Insurance Product Factory** repository - that can be drawn upon when launching new products, or versioning existing ones. You are free to use your product knowledge to decide the degree of similarity between the old and the new products and you have tools in place to test your creations.

Managing Products

The **Insurance Product Factory** makes it easier for you to streamline the development of insurance products, as well as their deployment and maintenance. This direct coupling along the product lifecycle makes place for better performance, also when it comes to portfolio management.

The products that you develop might transition between different business statuses - like **Draft**, **Approved**, **Closed**, or they might be updated to a new version, they might be the subject of special offers, or they might be part of recurring offers (but with limited availability), and so on. For all these cases and more, you use **Insurance Product Factory** to be on track with accurately recording and housing the said changes, control product behavior and, also, to efficiently organize the updated information.

Integrations

This solution is part of the Northstar and can be integrated with other components or solutions from **FintechOS**, allowing you to reap the resulting digital synergy. Few examples include:

- leveraging the capabilities of the Business Decisions Processor and Business Formulas, to implement complex decision modeling for insurance risk rules and premium calculations and apply them to different collections of insurance products.

- using the Digital Journeys functionality to expose your products to your potential customers.
- using the Digital Documents Processors to automatically generate personalized documents – including customized contracts and agreements, based on customer's input and the business document templates existing at the product level.
- using the Proposal Configurator on top of the **Insurance Product Factory** solution, in order to deliver a **Quote Configurator** customer experience - namely, allowing the eligible customers to review different insurance products, offers, or modules and configure their own insurance quote.
- using different insurance accelerators on top of the **Insurance Product Factory** solution, in order to speed up the delivery of specific insurance products.

More than that, for complex delivery projects, it is worth mentioning that the **Insurance Product Factory** can be also used in conjunction with different automation processors, that help insurers build their operations around the needs of their customers - such as Omnichannel Campaigns, or Hyper-Personalization Automation, and others. For more details, consult the Automation Blocks documentation.

IMPORTANT!
The **Insurance Product Factory** version 4.3.0 is supported by the **Innovation Studio** version **21.2.** and newer.

Business Pain Points

FintechOS clients use the **Insurance Product Factory** module to respond to different challenges related to:

- time-consuming routines;
- routines more prone to human error when done manually;
- the sheer volume of changes to be recorded and accounted for;
- little time to create insurance products and launch them;
- pressure to adapt constantly to fluctuating markets.

Insurance companies use **Insurance Product Factory** to reduce the completion time required to create, configure, activate, change, and manage insurance product portfolios.

Insurance Product Factory Key Features

The **Insurance Product Factory** module enables insurers to achieve efficiency, gain flexibility and organize their insurance products data for analysis and operational purposes easily. The solution has the following key features:

- Specific product configurations regarding availability periods for products, policy adjustments, policy transitions automated workflows or statement generation settings.
- Adjustable collections of insurance product items, additional clauses, modules and risks covered by the insurance products.
- Integration with Formula Engine for premium amount calculations, insurance risk rules and underwriting rules configurations.
- Product-related documents management.
- Full versioning functionality.
- Varying levels of access concerning users' involvement with a specific product, or product view.

Insurance Product Factory Key Benefits

The benefits of using **Insurance Product Factory** are the following:

- Digital: automates routines, improves accuracy, manages complexity and transaction volumes, in real-time.
- Flexible: manages one-time and low usage product launching scenarios and also the frequently recurring ones and it can scale from the simplest of product portfolio management models to the most complex ones.

- Safe: **FintechOS** role-based security architecture allows users, once authenticated, to interact only with the selected product, flow steps or interfaces associated with their role.
- Personalized: the solution is highly customizable - you can use **Innovation Studio** to further tailor **Insurance Product Factory** to respond to your particular requirements.

Insurance Product Factory Potential Users

This solution is aimed at everyday use by different types of insurance professionals:

- Product and portfolio development managers and executives.
- Other types of experts that might have a say in the product development - such as marketers and underwriters.
- Product and portfolio development analysts and consultants.
- Digital marketing agencies, with a strong footprint in the insurance field, seeking to build draft product offers, or MVPs, for their clients (insurance companies) - potentially as part of a larger deal that involves the market research phase, the product testing and the marketing of the respective product.
- IT companies or digital assets management agencies seeking to service a particular niche in the insurance field - such as small and medium sized insurance companies with strong local presence (and with legacy portfolios containing, for example, life policies underwritten years ago), currently looking to outsource the IT and administration of said policies (in order to improve customer satisfaction, lower costs, and even reduce capital requirements) but lacking the skills to go digital or scale with **FintechOS** just by themselves.

- Insurance company owners with an entrepreneurial mindset seeking to translate their monolithic legacy hard-coded insurance products into flexible mix-and-match highly personalized insurance products, ready to be delivered through different digital touchpoints, channels, or portals or just looking to re-engineer and evolve the products they bring to market, and making them ready for new types of distribution.
- Any consultancy agency with a strong footprint in the insurance field, that is specialized in digital product co-creation, adjustment and enrichment.

HINT

Integrate **Insurance Product Factory** with more **FintechOS** solutions for insurance in order to make the best of process automation for your company, portfolios, products and clients!

Managing Insurance Products

The **Insurance Product Factory** enables you to create, adjust and maintain insurance products, and product sub-components. When they are in **Active** status, your products can be exposed to potential customers, through different digital journeys.

Check the following pages to find out more about how you can use this solution:

Insurance Products - for details about how to create and configure insurance products.

Insurance Risks - for details about how to create and configure risks.

Insurance Types - for details about how to create and configure types.

Product Life Cycle - for details about product operations, product versioning and more.

> **IMPORTANT!**
> When building your first product, you must make sure that at least one Insurance Type is defined!

Insurance Products

The **Insurance Product Factory** enables you to create a great variety of products aimed to help potential customers get insurance, as effortlessly as possible.

For example, an insurer who wants to seize the opportunity, following a market boom, such as lots of owners selling their houses in a particular area, can create a new **Home Insurance Product** promptly. The premium amount can be calculated separately for each product item - such as the contents of the house, allowing a better understanding concerning pricing the whole product. If needed, the insurer can also

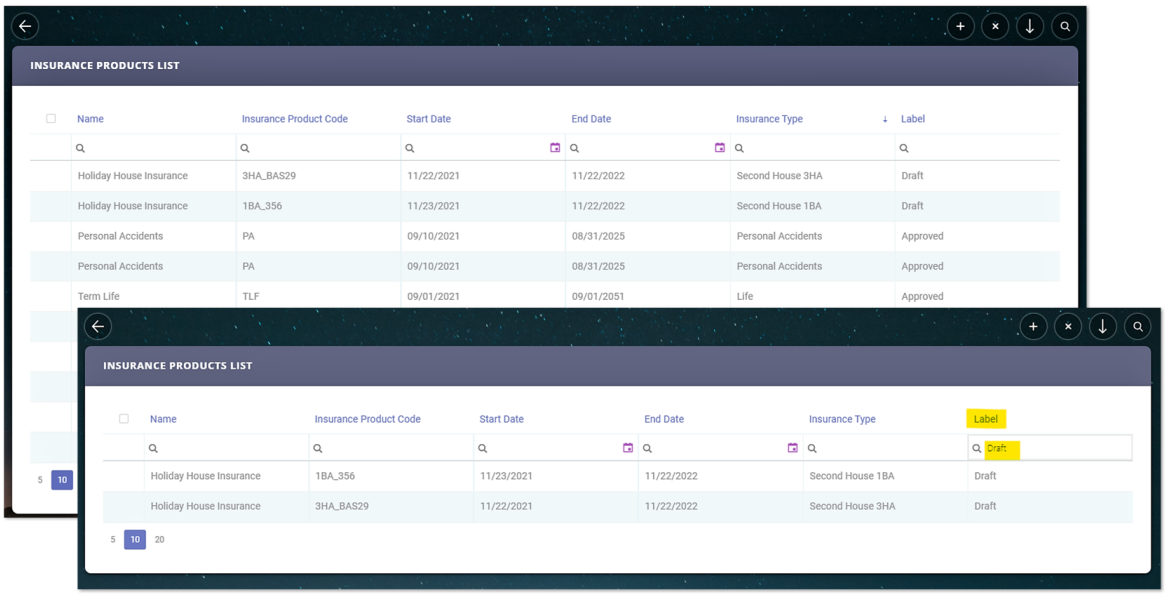
set a time limit for the product availability. Once activated, the product is digital journey-ready and the insurer can expose it to potential customers through different digital channels, relevant to that area.

> **IMPORTANT!**
> **Prerequisite**: In order to create an **Insurance Product**, at least one Insurance Type must be defined!

Insurance Products View

In **Innovation Studio**, in the **Insurance Products** section, you have an overview of all the insurance products created in your system. This is an all-inclusive view; yet, you can also search and sort your products for easier processing. For example, if you want to view all the products in **Draft** status, you can use the **Search by Label** option and sort all your invoices accordingly.

Below, you can see an example of a list view for products and a product search example.



Follow these steps to view your **Insurance Products**:

1. Go to **Innovation Studio**.
2. At the top left corner of your screen, click the main menu icon to open the main dropdown list.
3. From this main list, click **Insurance Product Factory**. A second dropdown opens.
4. Next, click **Insurance Products** to go to the **Insurance Products List**.

On the **Insurance Products List** page:

- To inspect a record from the grid, double-click it.
- To add a new record, click **Insert**, at the top right corner of the page.
- To edit a **Draft** record from the grid, double-click it and press **Edit**.
- To delete a record from the grid, select it and click **Delete**, at the top right corner of the page.

> **HINT**
> You can export one or more records by pressing **Export**, at the top right corner of your screen.

Creating an Insurance Product

At the top right corner of the **Insurance Products List** page, click **Insert** to add a new insurance product. The **Insurance Product** dynamic form opens. This form allows you to create your desired product and configure the product settings. To complete the creation journey for your product, go through all the tabs listed below. Filling in information in the first tab activates the next tabs and lets you move forward through the product creation journey. However, if you need to interrupt your journey, you can use the **Save and close** button to save your product as **Draft**, and complete the journey at a later time. While the product is in **Draft** business status, the form is still editable. For more details, see the **Creating a Draft Insurance Product** section, from the Product Lifecycle page.

Use the tabs described below to create your insurance product:

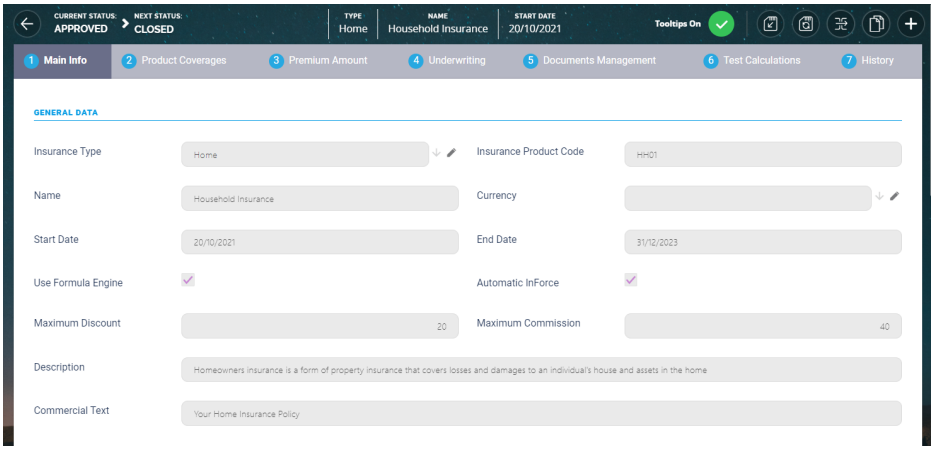
Main Info

The **Main Info** tab has two sections:

- The **General Data** section - holds details about the product's main characteristics, those which are most visible to the final customer.
- The **Product Configurations** section - lets you configure the details for the policy coverage, policy administration, the scheduling of payments and billing, and the management of claims.

> **NOTE**
> You can save and come back to edit your draft product as many times as you want. After you activate it, editing is done only by creating a new version of the product. For more details, consult the Product Lifecycle page.

General Data



In this section, generally describe your new product. The following fields are available:

Field	Description
Insurance Type	Click the dropdown to select a **Type** of insurance for your insurance product - ex. Auto, Health, Home, Travel. See Insurance Types for details.
Insurance Product Code	Use this text area to fill in the code of the insurance product.
Name	Insert the name of your insurance product.

Field	Description
Currency	From the dropdown list, select a currency for your insurance product.
Start Date	Pick the date when your product becomes available.
End Date	Pick the date when the availability of your product ends.
Use Formula Engine	Mandatory check for calculating different insurance formulas for your product. It allows the connection with FintechOS Business Formulas.
Automatic InForce	Set to true if you allow automatic transitioning from **Issued** to **InForce** business status for policies including this insurance product, when the date they are issued is the same as their begin date. See Policy Admin for more details.
Maximum Discount	Set the maximum percentage of the commercial discount which can be offered in the sales process for your product. This field allows numeric values.
Maximum Commission	Set the maximum percentage of the commission which can be offered to intermediary sellers. This field allows numeric values.
Description	Field for describing the insurance product (500 characters).
Commercial Text	Field for commercial text (300 characters).

After inserting the required information, move to the next section of the tab.

Product Configurations

In this section, describe your product configurations concerning policy coverage, policy administration, the scheduling of payments and billing, and the management of claims.

The following fields are available for configuring **Policy Coverage**:

Field	Description
Grace Period	The grace period for a policy. Input the number of time units for the grace period.

Field	Description
Grace Period Type	From the option set, choose **Days** or **Months** to indicate the type of grace period.
Total Indemnity Limit	Insert the maximum coverage amount provided per insurance policy.

The following fields are available for configuring **Policy Administration**:

Field	Description
Free Withdrawal Period Limit	Set the Free Withdrawal Period Limit - necessary for Cancellation processes.
Renew Type	Choose the type of the insurance policy renewal. The option set values are: **No** (default), **Automatic renewal**, and **Renewal offers**. If you choose **Automatic renewal**, the following fields become available: **Renewal Validity** - where the option set values are: **Yearly**, **Monthly**, and **SameValidity**. **Renewing Policy** - where the option set values are: **Same Policy** and **New Policy**. **Renewal Tariff** - where the option set values are: **Same tariff** and **Actual tariff**. **No of Days Before Renewal** - insert the number of days before the system notifies you about the coming renewal opportunity, for policies incorporating the specified product.
Suspension	Choose if you allow suspension settings on your insurance product. The option set values are: **No** (default) and **Yes**.
Type of Suspension	If you set Yes for Suspension, this is the field that allows you to select what type of suspension you allow for policies incorporating your product. The option set values are: **Generic** and **Specific**. Choose **Specific** if you want your particular settings to be included into any suspension flow for that product. Choose **Generic** if you allow general suspension settings for your product, managed through the **Policy Admin** module.

Field	Description
Days Before Suspended	Set the number of days before a policy enters the suspension flow.
Premium Update due to Suspension	Choose if you allow premium updates on the suspended policies. The option set values are: **No** and **Monthly**.
Max. no of consecutive suspended months before lapsing	Set the maximum number of consecutive suspended months before a policy is moved to the lapsing flow.
Prorata Type Configuration	Set the proportion rate type for premium payments. The option set values are: - **Generic** if you allow the generic setting to be used. - **Specific** if you want to set a specific rate. When you choose this option, the **Prorata Type** option set becomes available and you can choose between its values: **Daily** or **Monthly**.

The following fields are available for configuring **Payments Schedule & Billing**:

Field	Description
Payment Period Grace (days)	Insert the number of days for the payment grace period.

Field	Description
Write off	Set the tolerance threshold for writing off payments that fall shorter than the expected agreed installment amount (for example 74.5$ instead of 75$). The option set values are: **No write-off**, **Generic**, and **Specific Write Off**. Choose: **NoWriteOff** if no rule for writing off is applied for your product. **GenericWriteOff** if you allow the generic write-off settings to be used. **SpecificWriteOff** if you want to set the value for this parameter yourself. When you choose the Specific Write-Off option, a Json pop-up appears and you can add your own key-value pairs, per different currencies. Set the desired values for this parameter. Your inserts are automatically saved.
Statement Generation	Configure the values to be used for automatic invoice generation. The option set values are: **Generic**, **Specific SGDAY** and **Specific Day**. Choose: **Generic** if you allow the generic setting to be used for automatic invoice generation. **Specific SGDAY** if you want to set a specific day for invoice generation. When you choose this option, the **No. of Days in Advance (SGDAY)** field becomes available and you can insert your specific number. **Specific Day** if you want the automatic invoice generation to be performed by the system on a specific day of the month. When you choose this option, the **Specific day of the month** field becomes available and you can insert your specific number.

* **SGDAY** translates to Statement Generation Day - This parameter stores the number of days in advance (before the payment due date), for automatically generating an invoice for a scheduled payment (installment) on a policy.

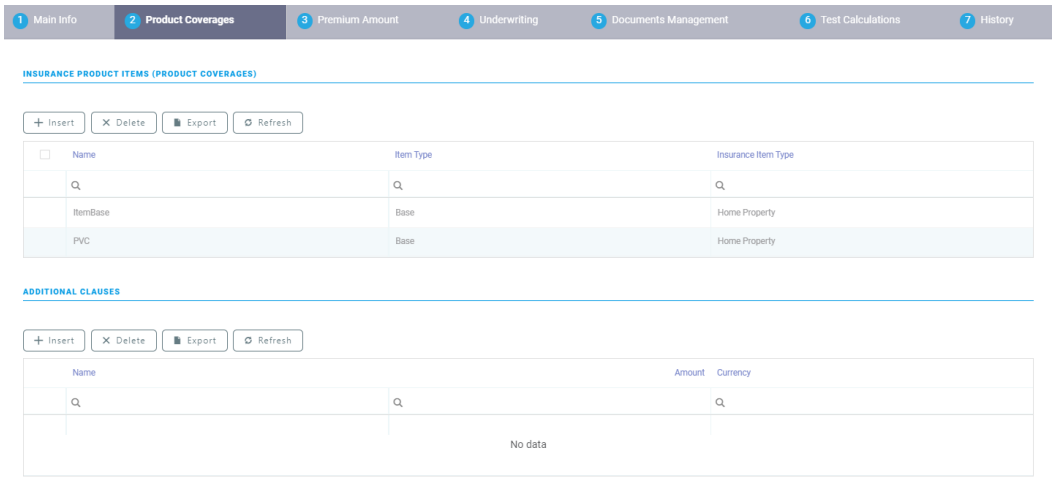
The following fields are available for configuring the **Claims Management**:

Field	Description
Claim Prescribing Period (months)	The period of prescription of a claim, expressed in months.
Claim Notification Period Limit (hours)	The period during which the claimant can notify the damage produced - in order to open a claim on a policy that contains this insurance product, expressed in hours.
Update Indemnity Limit	Check the box in order for the system to make automatic updates of the indemnity limit after every claim payment.

Click **Save and reload**. The other tabs become available and you can see them at the top of the screen.

Product Coverages

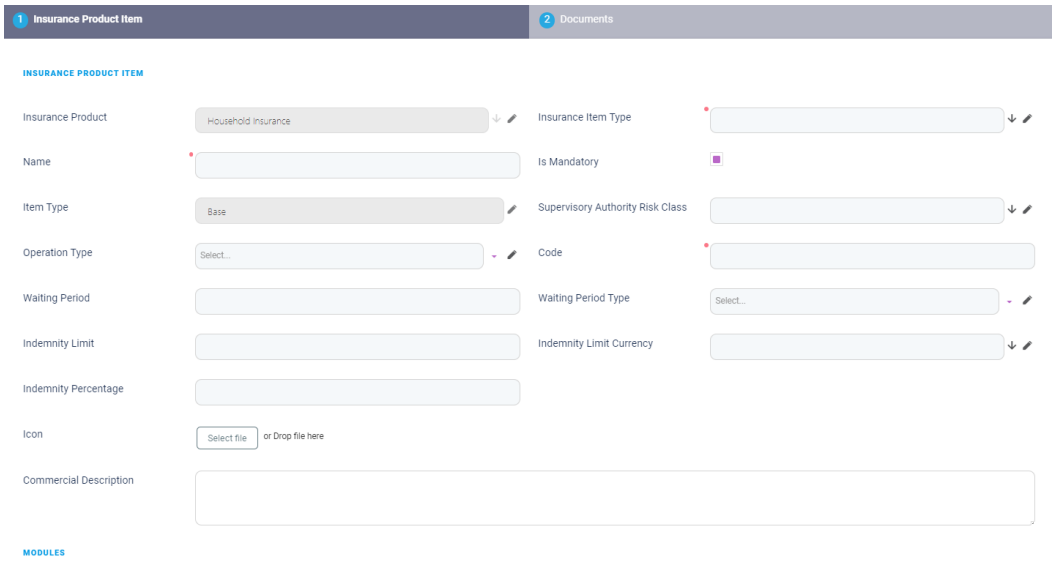
The **Product Coverages** tab allows you to configure the coverage for each product item. For example, for property insurance, a customer might buy coverage for two types of product items: first, the house and second, the contents of the house - which can be covered individually, also.



This tab has two sections:

- **Insurance Product Items** - This section is reserved for items of type **Base**. For example, for a **Property Insurance** policy, the **Base** insurance product item could be the coverage for the actual building, only. And the coverage for the contents of the house could be included as a **Rider**, see below.
- **Additional Clauses** - In this section, you can include **Riders** - additional risks that the customer wants to cover. For example, for a Life Insurance policy, additional coverage may potentially refer to losing working capacity.

Proceed to insert a **Base** product item. In the header section of the form, under **Insurance Product Items**, click **Insert**. The **Insurance Product Item** window opens. This form allows you to add any number of base items attached to your insurance product and any relevant documents describing each added item.



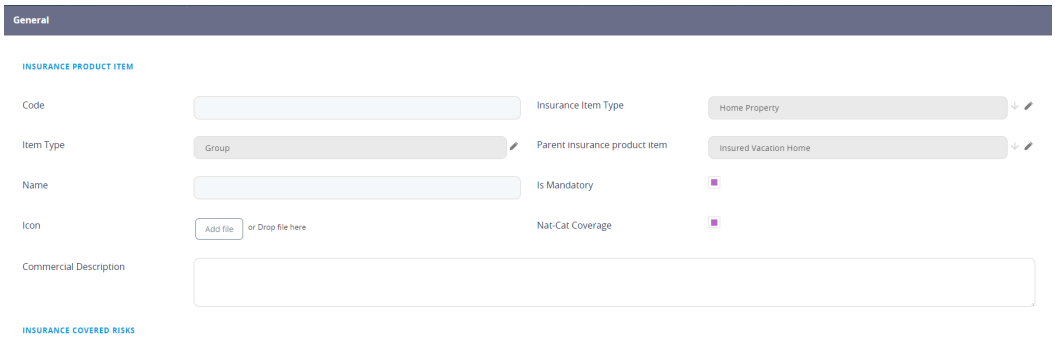
The following fields are available for configuring each **Product Item** that you create:

Field	Description
Insurance Product	The **Insurance Product** to which the current **Product Item** is assigned to. This information is automatically filled in by the system.

Field	Description
Insurance Item Type	From the dropdown, select the **Type** of the insurance product item (ex. House/ Car). If you need to add a new **Item Type**, see the Insurance Types page
Name	Add the name of the insured product item.
Is Mandatory	Check the box if the item is mandatory. For example, a mandatory house insurance as part of a mortgage.
Item Type	**Base** - It is automatically filled in by the system.
Operation Type	Choose **Sum** to have this product item included into the calculation of the final premium, for the current product.
Supervisory Authority Risk Class	This is a dropdown with a list of risk classes, as reported by insurers to their local, regional or national supervisory authority. Choose a risk class that covers your insurance product item.
Code	The code of the product.
Waiting Period	The amount of time a policyholder must wait before the coverage for the current item comes into effect. Insert the number of time units for the waiting period (until the current coverage becomes available on a policy).
Waiting Period Type	From the option set, choose **Days**, **Weeks** or **Months** to indicate the type of waiting period.
Indemnity Limit	The monetary amount of the coverage provided under the insurance policy.
Indemnity Limit Currency	From the dropdown list, select a currency for the indemnity limit of your insurance product item.
Indemnity Percentage	The percentage of indemnity limit, set-up as percentage from the total indemnity limit of the current product.
Icon	Upload an icon for the product item, if the case.
Commercial description	Text area for describing the product item.

Click **Save and reload**. Next, the **Modules** grid becomes available, at the bottom of the form. This grid allows you to correlate groups of risks coverage (eg. natural disasters) with your current **Insurance Product Item**.

In the **Modules** grid, click **Insert** to open the **General** window.

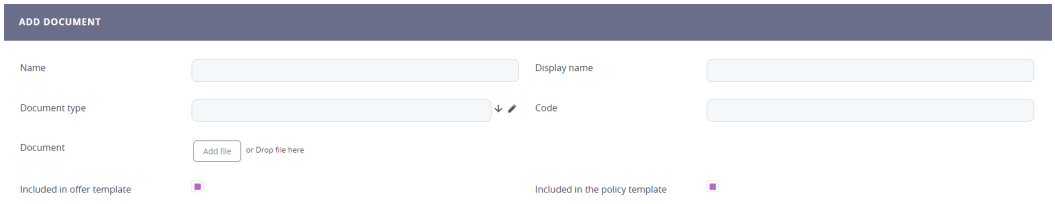


For each **Risk Group** that you add, the following information must be provided:

Field	Description
Code	The Code for your document.
Insurance Item Type	It is automatically filled in by the system with the current item type.
Item Type	It is automatically filled in by the system with the type **Group**.
Parent Insurance Product Item	It is automatically filled in by the system with the **Parent Insurance Product Item**.
Name	Add the name for the **Risk Group**.
Is Mandatory	Check the box if the feature is a mandatory one.
Nat-Cat coverage	Check the box if the risk item belongs to the Natural Catastrophes group.
Icon	Upload an icon for the risk item, if necessary.
Commercial description	Text area for describing the product item.

Adding these details about the groups of risks coverage for your current item unlocks a view, at the bottom of the form, in which you can see all the types of risks covered by the **Insurance Product** that you are currently configuring. You can explore the list of risks and if necessary, you can edit the risk types from the Insurance Risks menu.

Click **Save and reload** and then click **Documents**. This window allows you to add any relevant documents related to the product item. For each **Document** that you upload, the following information must be provided:

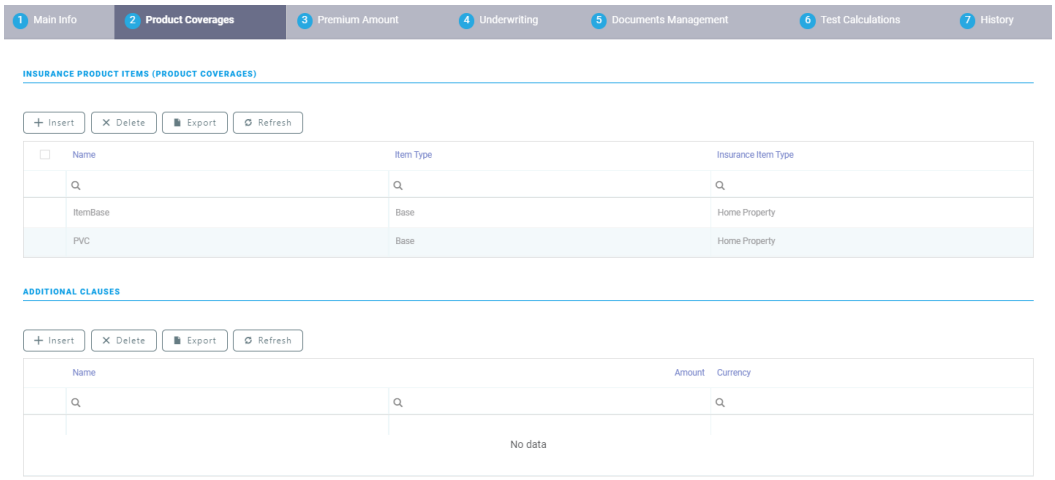


Field	Description
Name	The name of your document.
Display name	The display name of your document.
Document type	From the option set, choose between **Policy**, **Terms & Conditions** or**IPID** - Insurance Product Information Document - which is a type of document presented during the **Quote&Apply** flow.
Code	The code for your document.
Included in offer template	Check the box if your document must be included in the product offer template.
Included in the policy template	Check the box if your document must be included in the policy template.

Click **Add File** to upload your document.

Add as many base **Product Items** as you need for the current **Insurance Product**. Use the **Modules** grid on every **Product Item** to set group risks, the same way you did for the first base **Product Item**, if the case.

Once finished, click **Save and close** and navigate back to the **Insurance Product Items** window. Here, you can use the **Additional Clauses** grid to add **Rider** items for your insurance product.



Under **Additional Clauses**, click **Insert**. The **Insurance Product Item Rider** window opens. This form is similar to the form for the **Base** insurance product items, the exception being the **Item Type** field - which is automatically filled in by the system with the **Rider** option.

The following fields are available for configuring each **Product Item Rider** that you create:

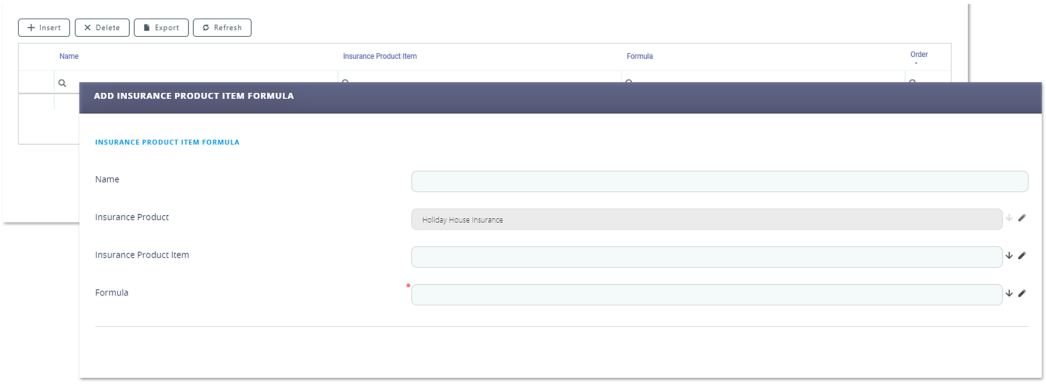
Field	Description
Insurance Product	The **Insurance Product** to which the current **Product Item** is assigned to. This information is automatically filled in by the system.
Insurance Item Type	From the dropdown, select the **Type** of the insurance product item (ex. House/ Car). If you need to add a new **Item Type**, see the Insurance Types page.
Name	Add the name of the insured product item.
Is Mandatory	Check the box if the item is mandatory. For example, a mandatory house insurance as part of a mortgage.
Item Type	**Rider** - It is automatically filled in by the system.
Operation Type	Choose **Sum** to have this product item included into the calculation of the final premium, for the current product.
Supervisory Authority Risk Class	This is a dropdown with a list of risk classes, as reported by insurers to their local, regional or national supervisory authority. Choose a risk class that covers your insurance product item.
Code	The code of the product.

Field	Description
Waiting Period	The amount of time a policyholder must wait before the coverage for the current item comes into effect. Insert the number of time units for the waiting period (until the current coverage becomes available on a policy).
Waiting Period Type	From the option set, choose **Days**, **Weeks** or **Months** to indicate the type of waiting period.
Indemnity Limit	The monetary amount of the coverage provided under the insurance policy.
Indemnity Limit Currency	From the dropdown list, select a currency for the indemnity limit of your insurance product item.
Indemnity Percentage	The percentage of indemnity limit, set-up as percentage from the total indemnity limit of the current product.
Icon	Upload an icon for the product item, if the case.
Commercial description	Text area for describing the product item.

Click **Save and Reload**. Add as many **Product Item Riders** you need for your **Insurance Product**. Use the **Modules** grid on every **Product Item Riders** to set group risks, the same way you did for the **Product Items**, if the case.

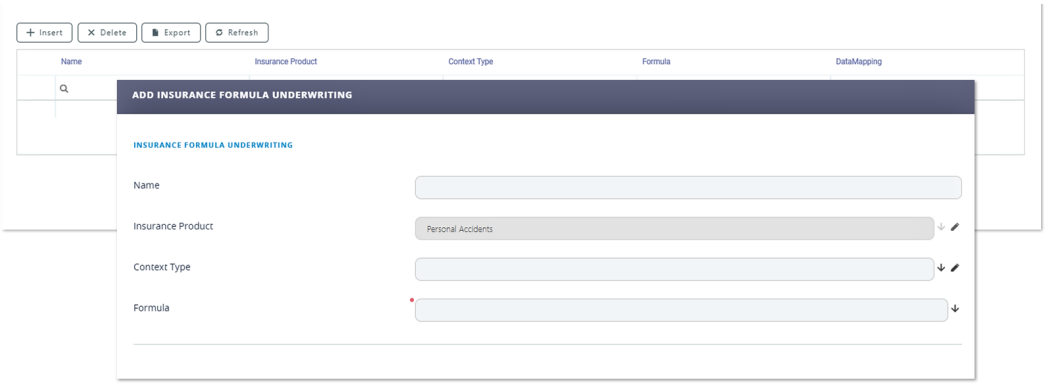
Premium Amount

This section allows you to use different formulas in order to calculate the **Premium Amount** for each **Item** attached to your current **Product**. For more information about creating your own formulas and examples of the logic used for calculating premium amounts for insurance products, start from the Business Formulas Integration page.



Underwriting

This section allows you to use different formulas for analyzing different risks related to property insurance and produce a status or decision regarding the policy - for example, establishing whether the proposed risks are insurable for that particular property. You can also configure formulas for determining whether the policy proceeds further with an automatic flow or if it needs to go on a manual approval flow. For more information about creating your own formulas and examples of the logic used for risk scoring and underwriting for insurance products, start from the Business Formulas Integration page.



Documents Management

This section allows you to manage the documents that are associated with your product. For example:

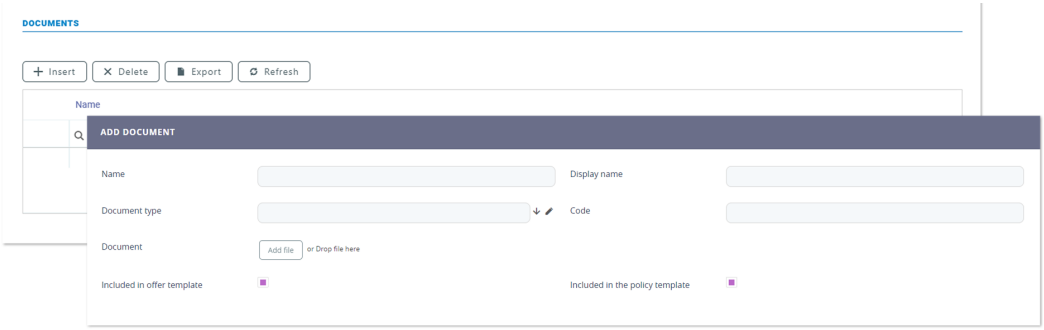
- terms and conditions for the current insurance product,
- the mandatory general presentation of the product as requested by the authorities,
- mandatory clauses to be included in contracts based on different criteria,
- sample presentations that are to be used in the quote & apply journey or for different customer personas, and more.

For each **Document** that you upload, the following information must be provided:

Field	Description
Name	The name of your document.

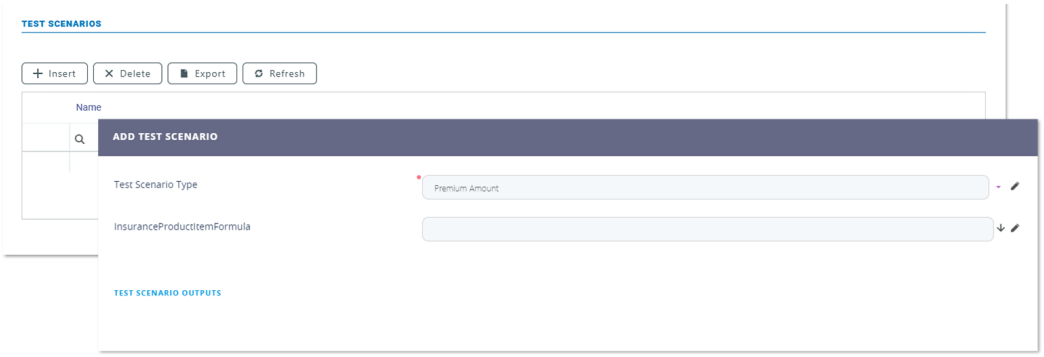
Field	Description
Display name	The display name of your document.
Document type	From the dropdown, choose between **Policy**, **Terms & Conditions** or**IPID** - Insurance Product Information Document - which is a type of document necessary for the **Quote&Apply** journeys.
Code	The Code for your document.
Included in offer template	Check the box if the document must be included in offer template.
Included in the policy template	Check the box if the document must be included in the policy template.

Click **Add File** to upload your document and then click **Save and reload** to continue your journey.



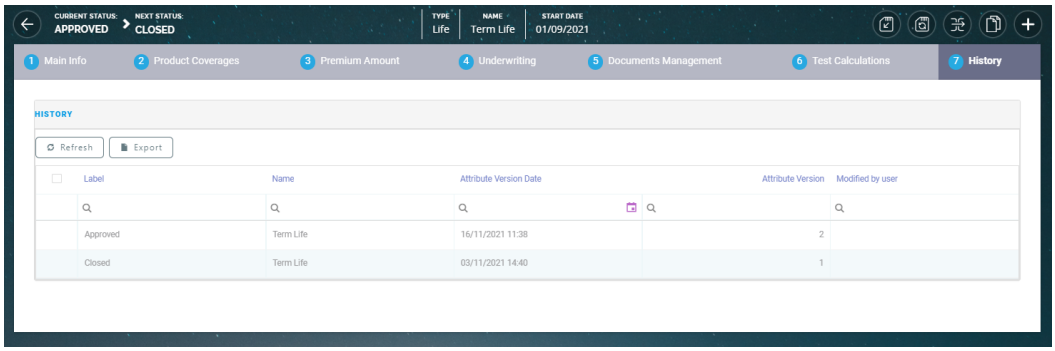
Test Calculations

This section allows you to test the **Premium Amount** and **Underwriting** formulas and see the results you get with different **Scenarios** regarding your **Insurance Product**. Using scenarios to test your formulas helps with fine-tuning and improving the general performance of your product.



History

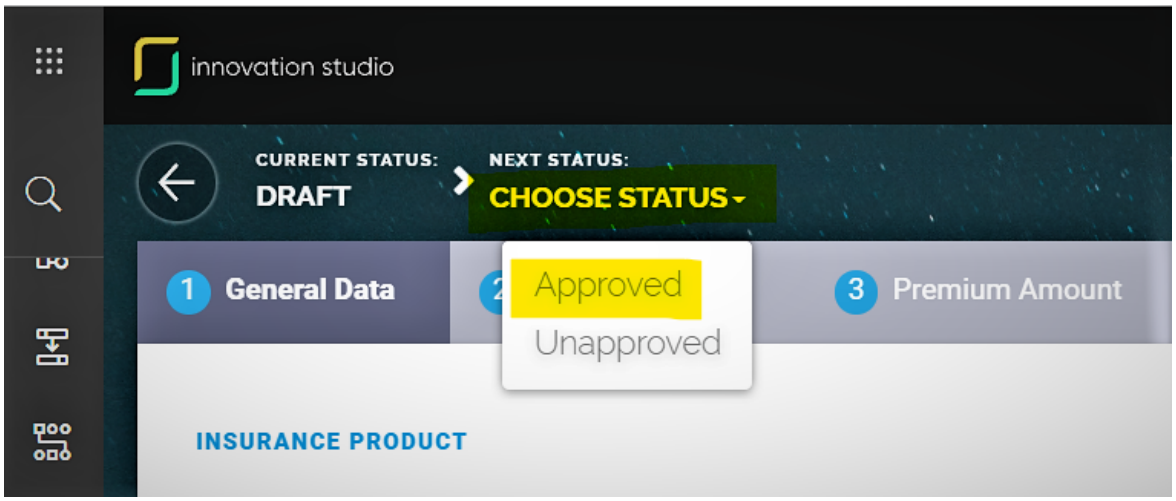
A history of the product allows you to understand how your product evolved over time and decide on updating its course, eventually. You go to the **History** tab when you need to have an overview over the product's life cycle, inform yourself about previous product versions (approved or unapproved), their workflow status, or to obtain details about users who modified the product.



NOTE

When you finished filling in the data, **you must activate your product** from the header option set. **Product status** determines if your insurance product is used in digital journeys targeting potential customers. In order to make it available for different digital journeys, the product should be in the **Approved** status. For details, see the Product Life Cycle page.

Below is an example of changing the product status, by using the status picker.



Insurance Risks

The **Insurance Product Factory** allows you to set up **Insurance Risks** independently from **Products**, so that they can be used in conjunction with multiple Insurance Products. An **Insurance Risk** - e.g. Earthquake, Car accident, Tornadoes, Theft, Death, Disability - informs about the risk coverage of a particular insurance product that you want to create. **Insurance Risks** can be used in conjunction with the Business Decisions Processor to implement complex decision modeling for insurance risk rules.

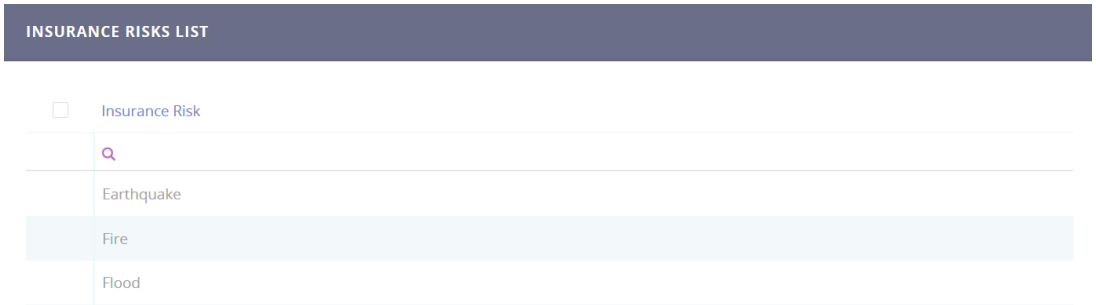
For example, the age of the building is a risk for a house insurance and this influences the decision whether to insure it or not. If the building is older than 80 years then a manual approval step might be needed. If the building is older than 100 years then the application might be rejected.

From the business perspective, **Insurance Risks** also make it easier to gather data for reporting & analysis.

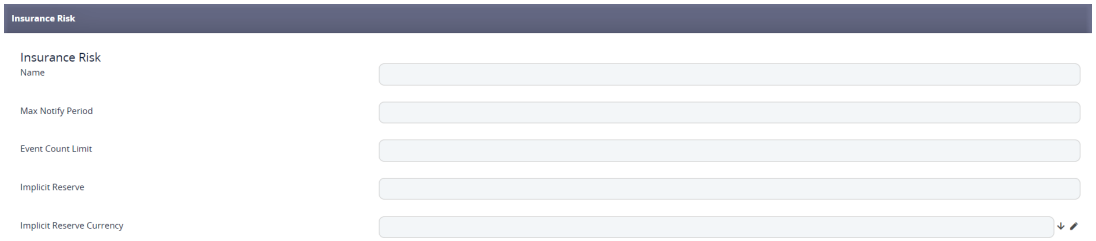
Creating Risks

In order to create an **Insurance Risk**, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Risks** to open the **Insurance Risks List** page.



5. At the top right corner of the page, click **Insert** to add a new insurance risk. The **Insurance Risk** form allows you to configure the insurance risk settings.



For each Risk that you create, the following information must be provided:

Field	Description
Name	Add the name of the insurance risk.
Max Notify Period	Add the maximum period for notification of risk.
Event Count Limit	Add the number of events covered by the policy.
Implicit Reserve	Provide the amount of the prudential reserve to be deposited for the current policy.
Implicit Reserve Currency	Add the currency of the prudential reserve.

Modifying Risks

In order to modify an **Insurance Risk**, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Risks** to open the **Insurance Risks List** page.
5. From the list, choose the desired **Insurance Risk** record and double-click it to open the edit form. Use the form to make your adjustments.
6. Click **Save and close**, at the top right corner of your screen.

Deleting Risks

To delete an **Insurance Risk** from the grid, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Risks** to open the **Insurance Risks List** page.
5. From the list, select the **Insurance Risk** record that you want to delete and then click **Delete**, at the top right corner of the page.

Insurance Types

The **Insurance Product Factory** allows you to set up **Insurance Types** independently from **Products**, so that they can be used in conjunction with multiple "Insurance Products" on page 10. An **Insurance Type** is mandatory in order to further assembly any kind of **Insurance Product**, as it informs about the main class of insurance product that you want to create - e.g. Auto, Health, Home, Travel. If needed, you can also use Types to further segregate between your Products - e.g. "Auto+Car type+Car year" (A_diesel_2000s).

From the business perspective, **Types** also make it easier to gather data for reporting & analysis.

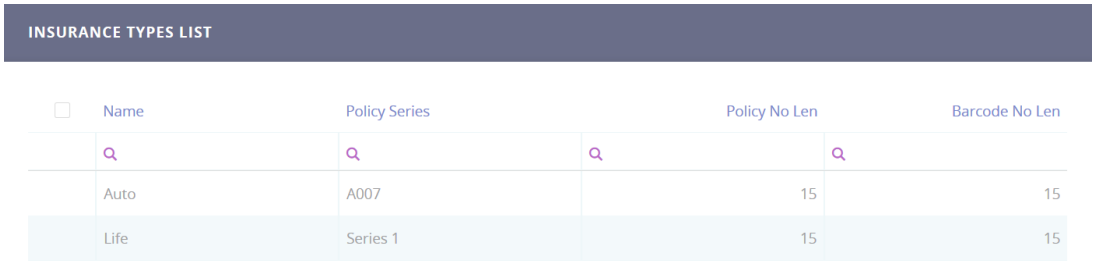
Creating Types

In order to create an **Insurance Type**, please follow the next steps:

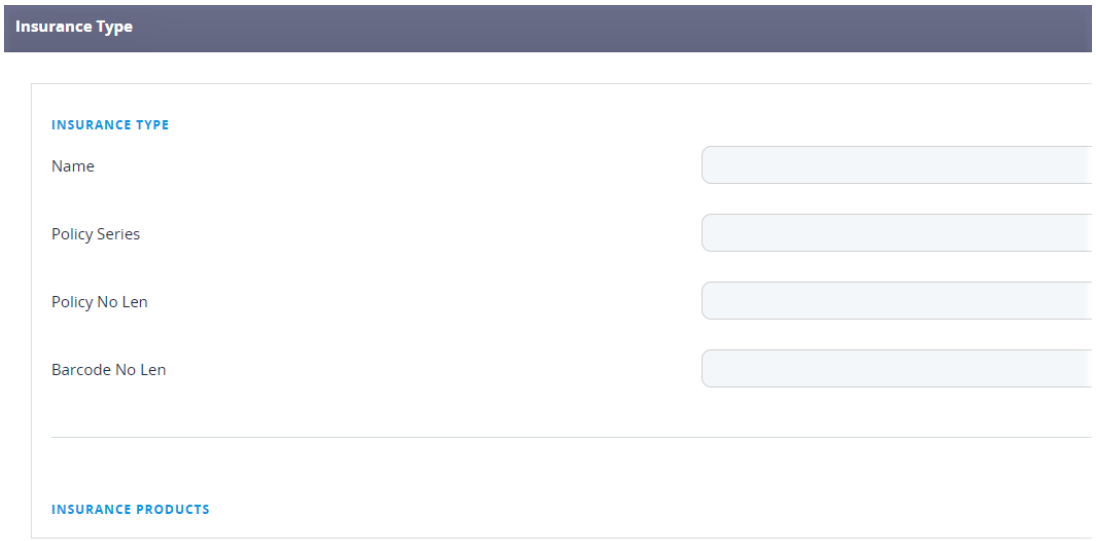
1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.

3. From the list, click **Insurance Product Factory**. A second dropdown opens.

4. Click **Insurance Types** to open the **Insurance Types List** page.



5. At the top right corner of the page, click **Insert** to add a new Type. The **Insurance Type** form allows you to configure the insurance type settings. Provide a name for the Type that you create.



Here is a description of the **Insurance Type** form:

Field	Description
Name	The name of the insurance type.
Policy series	No longer needs to be completed - the series of the insurance policy is presently configured through a sequencer.
Policy no len	No longer needs to be completed - the number of digits of the insurance policy number is presently configured through a sequencer.
Barcode no len	Not used in the configuration process - the number of digits of the insurance type's bar code number.

NOTE
The **Insurance Products** grid at the bottom of the **Insurance Type** form offers a view over all the Products associated with that particular Type. Inserting a new entry into the grid creates a new **Insurance Product** with the **Insurance Type** field pre-filled with the details from that specific Type. In some cases, this grid can be used as a shortcut from the Type to the Product form driven flow since clicking **Insert** on the Type grid moves the user to the first tab of the Product creation flow. For details, see Insurance Products.

Modifying Types

In order to modify an **Insurance Type**, please follow the next steps:

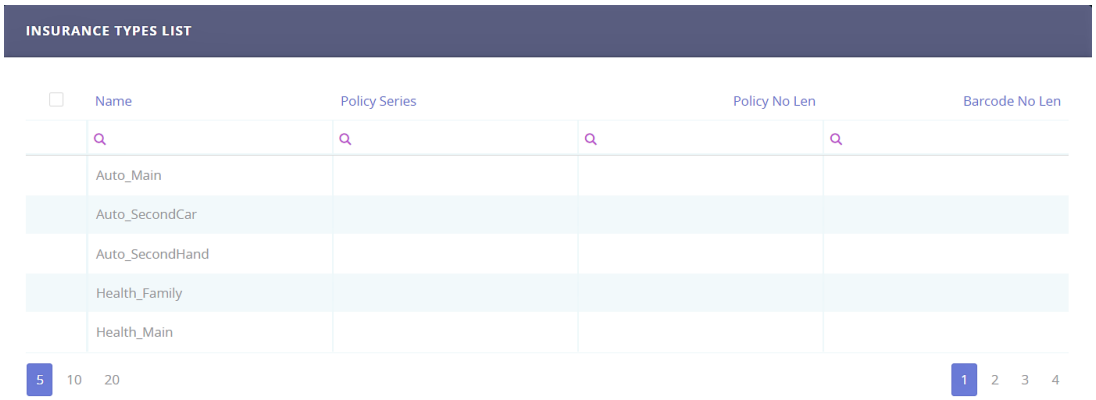
1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Types** to open the **Insurance Types List** page.

5. From the list, choose the desired **Insurance Type** record and double-click it to open the edit form. Use the form to make adjustments.
6. Click **Save and close**, at the top right corner of your screen.

Deleting Types

To delete an **Insurance Type** from the grid, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Types** to open the **Insurance Types List** page.



5. From the list, tick the desired **Insurance Type** record and then click **Delete** at the top right corner of the page.

> **IMPORTANT!**
> You can promptly modify or delete **Insurance Types** using the low-code capabilities of **Innovation Studio**. However, before modifying a certain Type, take into account the insurance products already attached to that Type.

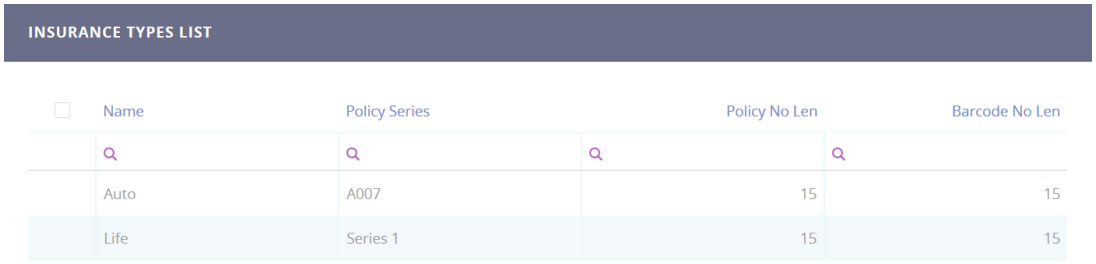
Policy Alteration Types for Insurance Types

There are times when you need to alter a policy - for example, if the policyholder decides to pay through an online payment processor instead of direct debit, an update of the payment type must be made on the policy. In **Insurance Product**

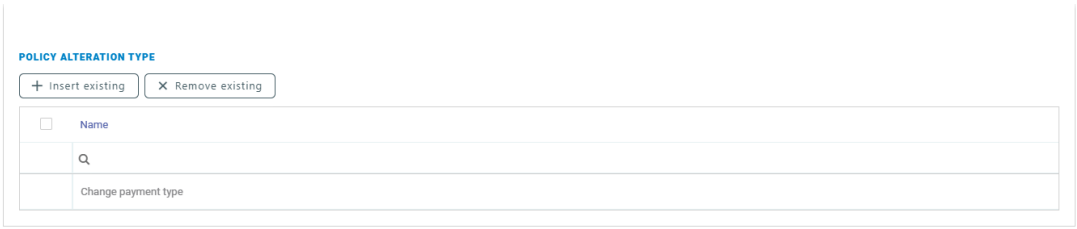
Factory, you use the **Policy Alteration** option to perform such updates.

In order to insert an **Alteration Type**, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Types** to open the **Insurance Types List** page.



5. From the list, choose the desired **Insurance Type** and double-click it to open the **Insurance Type** form.
6. Scroll down to the **Alteration Type** grid.



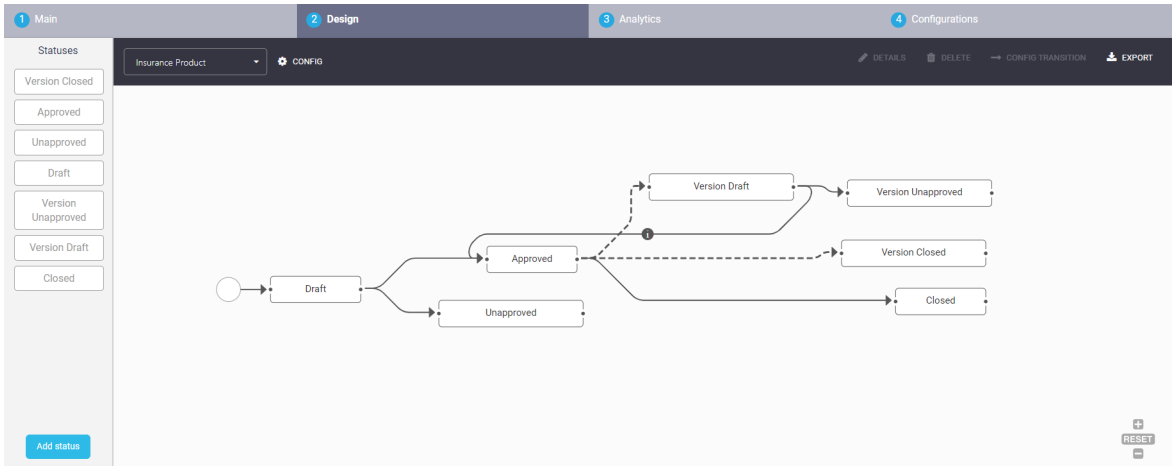
7. Use the form to **Insert** an **Alteration Type** for the policies belonging to that Insurance Type.
8. Press **Insert existing** and choose an alteration from the list and click **Ok**. You can select from the already configured options - **Change frequency**, **Change payment type**, or **Update coverage**.

Product Life Cycle

The **Insurance Product Factory** solution handles different types of changes affecting an insurance product during its lifetime. This is accomplished by transitioning the product through different business states (from **Draft** to **Closed**) and also by versioning it. The solution also logs these product life cycle details - in the product history tab, for further analytical use.

Product versioning is performed by using the **FintechOS** FTOS_INS_ VersioningWorkflow. This is a master business workflow, used to add the versioning functionality on any desired entity - including the entities used by the Northstar. For more details, scroll down to the Versioning an Insurance Product section.

Below is an example of the **FTOS_INS_VersioningWorkflow** business workflow used by the **Insurance Product Factory** solution, as it is displayed on **Innovation Studio**:



Product Behavior

The following are the behaviors - characteristic to any product, managed by this solution:

- Every product (or product version) starts in **Draft** status and must go through an approval process before going live.

- Once a product is live, its settings can no longer be modified.
- If you want to update a live product, you must create a new product version.
- When you create a new product version, the current version is retired.
- Only one version of a product can be live at one time.
- Only one draft version can be active at one time. If the case, you must close the current draft version and then, open another one.

Product States and State Transitions

States Descriptions

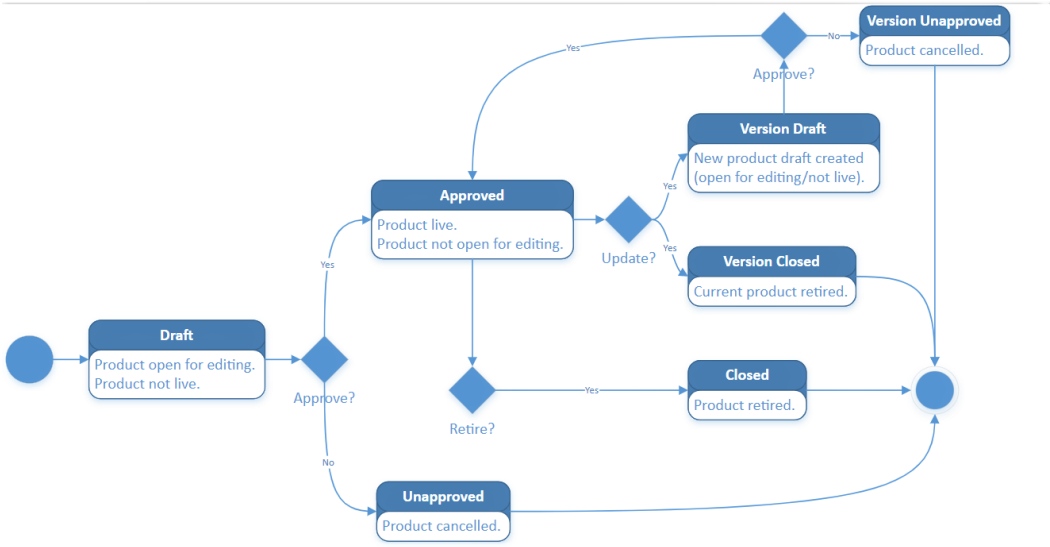
Status name	Description
Draft	Initial state for any insurance product registered in the system. The draft products can be edited.
Unapproved	When a draft product is canceled.
Approved	Ongoing state - for live products. Products can't be edited.
Closed	Final state. The product is retired. The product cannot be moved from this state to any other states.
Version Draft	When the selected product becomes editable.
Version Unapproved	When the updates on the product are not approved.
Version Closed	When the product's draft version is closed.

Product State Transitions

Transition	Description
_Draft	Initial state - for any insurance product issued into the system.
Draft_Approved	When a product goes from draft to live status. Live products can't be edited.
Draft_Unapproved	When the proposed product is canceled.
Approved_Closed	When a live product is no longer active.
Approved_Version Draft	When a new version is opened for a live product. The version is editable.
Approved_Version Closed	When the opened version is closed.

Transition	Description
Version Draft_ Approved	When the opened version is approved and becomes the new version for the selected product.
Version Draft_ Version Unapproved	When the opened version is not approved. The product goes live without being modified.

The product status transitions are illustrated below:



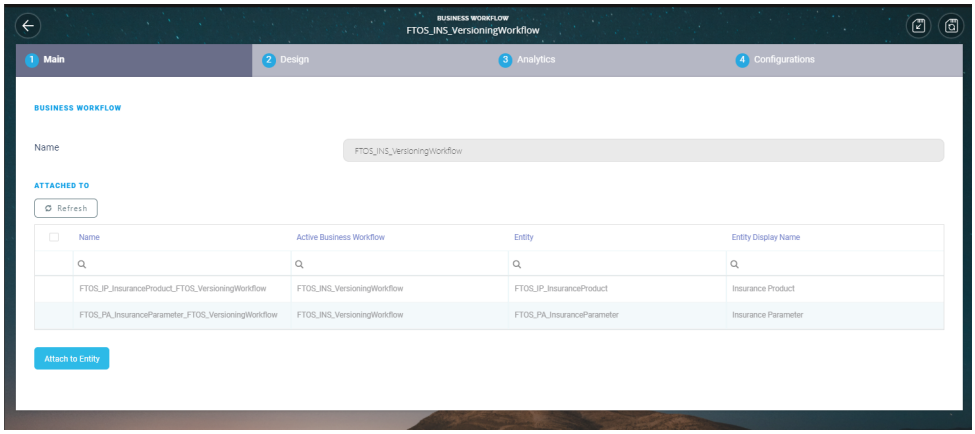
Business Workflow

To access the **Insurance Product Factory** versioning business workflow in **Innovation Studio**, take the following steps:

1. In **Innovation Studio**, navigate down the main menu to **Automation Blocks**.
2. Click **Automation Blocks** and next, click Business Workflows, to open the **Business Workflows Menu**, on the left.
3. Click **Business Workflows Designer** to open the **Business Workflows List** page.
4. From the list, select and double-click the **FTOS_INS_ VersioningWorkflow** record to inspect the business workflow attached to the Insurance Product entity.

When the form opens, you can see the main information about the workflow, its statuses and transitions.

Below is an example of the business workflow details, as they are displayed in **Innovation Studio**:



Product Operations

Besides creating products or product sub-components from scratch (like products, risks, or types), the following operations are also available for managing your product portfolio, in **Innovation Studio**:

Creating a Draft Insurance Product

There are times when you need to create a draft product, rapidly. While the product is in **Draft** business status, the form is still editable and you can complete the product creation journey at a later time.

To create a draft product, please follow the next steps:

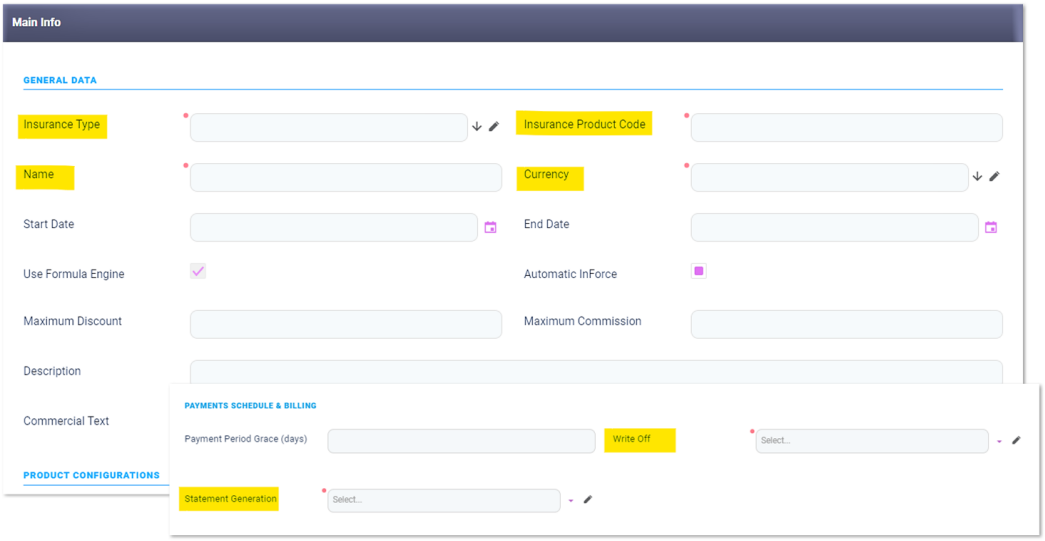
1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Products** to open the **Insurance Products List** page.

5. To add a new record, click **Insert**, at the top right corner of the page.
6. The **Insurance Product** form opens. Start filling in the mandatory fields, in order to be able to save the record as a **Draft**.

Mandatory fields:

Field	Description
Insurance Type	From the dropdown, select an **Insurance Type** for your insurance product - for example, Auto, Health, Home, or Travel. See Insurance Types for details.
Insurance Product Code	Fill in the code of your insurance product.
Name	Insert the name of your insurance product.
Currency	From the dropdown, select the currency for your insurance product.
Prorata Type Configuration	Set the proportion rate type for premium payments. More details about Prorata Type Configuration in the description of the **Policy Admin** configurations section.
Write off	Set the tolerance threshold for writing off payments. More details about Write off in the description of the **Payments Schedule & Billing** configurations section.
Statement Generation	Configure the values to be used for automatic invoice generation. More details about Statement Generation in the description of the **Payments Schedule & Billing** configurations section.

Below is an example of an **Insurance Product** form, with the mandatory fields highlighted:



> **NOTE**
> Cloning a product is also a way to create a draft product from scratch, since every cloned product is registered in the system only in **Draft** status.

To create a clone product, please see the next section.

Cloning an Insurance Product

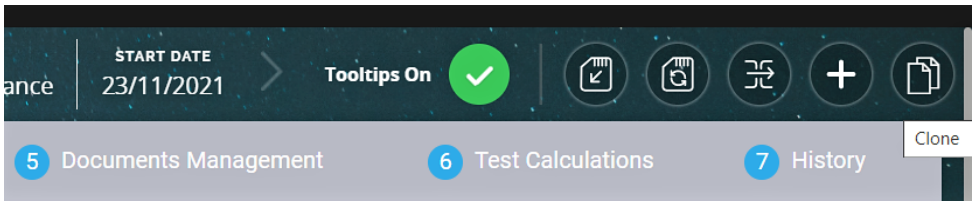
> **NOTE**
> Every **cloned product** is registered in the system in **Draft** status, irrespective of the status of the original product. While the product is in **Draft** business status, the form is still editable and you can complete the product creation journey at a later time. For more details, see also the Cloning Insurance Products page.

Here are the details for cloning an insurance product:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.

4. Click **Insurance Products** to open the **Insurance Products List** page.
5. From the list, select the insurance product that you want to clone and double-click it. Once opened, go to the top right corner of the screen, and press the **Clone** button to launch the cloning.

   Below is an example of how the button looks like, in **Innovation Studio**:



6. Next, the cloning pop-up opens. Use it to fill in a **Name** and an **Insurance Product Code** for the new product. After cloning, the product view opens and you can edit your new product.

Modifying an Insurance Product

> **NOTE**
> When your Product is in **Active** status, changing it may be done only by adding a **new version**. For this case, you must follow the details from the next section, below.

Here are the details for modifying an insurance product that is in **Draft** status:

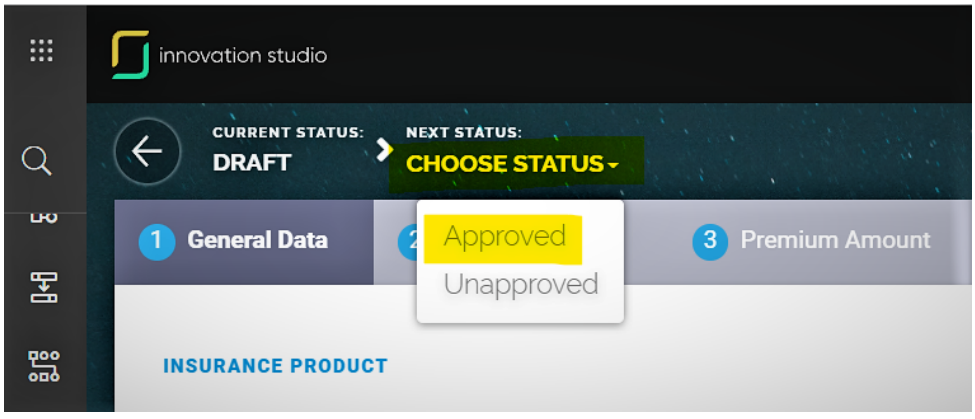
1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Products** to open the **Insurance Products List** page.

5. From the list, select the insurance product that you want to edit and double-click it. Once opened, use the form to edit your product.

Versioning an Insurance Product

Here are the details for versioning an insurance product:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Products** to open the **Insurance Products List** page.
5. From the list, select the **Insurance Product** that you want to version and double-click it. Once opened, go to the top right corner of the screen, and press the **Plus** button to add a new version of the product.
6. Use the form to edit your **Product**.
7. Once finished, change the status of the version from **Version Draft** to **Approved**, at the top left corner of the screen.
   Below is an example of changing the product status, by using the status picker.



8. After versioning, the product view opens and you can see your adjustments.

   You can also check the versioning log in the product's **History** tab.

> **NOTE**
> For deploying changes on existing products, you can also choose to import the changes in a **Data Config Definition** file, as this functionality has the versioning mechanism embedded. For more details, see also the Importing Insurance Products page.

Deleting an Insurance Product

> **IMPORTANT!**
> Pay attention to the fact that some policies might include the product that you want to delete.

To delete an **Insurance Product** record from the grid, please follow the next steps:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Insurance Product Factory**. A second dropdown opens.
4. Click **Insurance Products** to open the **Insurance Products List** page.
5. From the list, select the record that you want to delete and then click **Delete**, at the top right corner of the page.

Importing Product Data

This functionality lets you add product data on a destination environment by using the **FintechOSData Import Templates**. The most important use case for this functionality is a fast bulk import into the system of numerous new insurance products - with all imported products being registered in **Draft** status. But this is not the only use case. For more details, consult the

Importing Insurance Products page.

Here are the details for importing **Product Data** by using the available **Data Import Templates** configured for the **Insurance Product Factory** solution:

1. Open **Innovation Studio**.
2. At the top left corner, click the main menu icon to open the dropdown list.
3. From the list, click **Evolutive Data Core**. A second dropdown opens.
4. Click **Data Import Templates** to open the **Data Import Templates List** page.
5. From the list, select the **Data Import Template** that you want to update and double-click it. Once opened, scroll down to the **List Of Data Imports** section, and press **Insert** to upload your file. After import, you can see the new import added in the list and you also check the versioning log in the product's **History** tab.

IMPORTANT!
Templates are different depending on what kind of data needs to be added. Check the Importing Insurance Products page, for more details about the templates for every import category.

HINT
The **Insurance Product Factory** maintains the underlying data in a consistent data model. Consequently, you have a reliable data model for your insurance products that you can reference when you build your digital journeys. This allows you to manage your product portfolio at will, without having to re-code your digital journeys every time an insurance product is added, updated, or retired.

Installing Insurance Product Factory 4.3.0

Follow the guidelines below to import and configure the **Insurance Product Factory** module.

1 Import Packages

Digital assets location: **FintechOS** Release Hub

The following are the packages that need to be imported (scroll down for their descriptions):

- **Insurance Product Factory v4.3.0** is the digital asset designed for managing the entire insurance portfolio.
- **Insurance Product FactoryImport Formulas v4.2.0** contains the formulas used by the demo insurance products.
- **Insurance Product Factory Import Package v4.2.0** is the data import configuration package for insurance products, insurance risks, and insurance types. It also contains demo insurance products and versioning settings. See details below.

For more details about every digital asset, see also the Digital Assets page.

2 Installation

1. In **Innovation Studio** navigate down the main menu to the **DevOps** menu entry.

2. Click **DevOps**. From the dropdown, click on **Deployment Package** to open the **Deployment Package List** page.

3. On the top right corner, click the **Import deployment package** icon. The local **File explorer** window opens.

4. Perform the installation as described in the standard procedure.

3 Packages Descriptions

Insurance Product Factory v4.3.0

Digital asset description: The Insurance Product Factory is the digital asset used for managing the entire insurance portfolio. This is the first asset to be installed in the entire Northstar.

Platform version compatibility: High Productivity Fintech Infrastructure 21.2.2.

Dependencies: SySDigitalSolutionPackages v21.2.2000 or higher.

Insurance Apps Dependencies: N/A.

Upgrade from: Insurance Product Factory v2.14.9

Upgrade instructions: Before importing the digital asset, run the sql scripts available in the **Upgrade** folder from the digital asset location.

Install instructions: follow the standard procedure on how to import a digital asset.

After installment configurations: N/A.

Insurance Product Factory Import Formulas v4.2.0

Digital asset description: Formulas used by the insurance demo products.

Platform version compatibility: High Productivity Fintech Infrastructure 21.2.2

Dependencies: SySDigitalSolutionPackages v21.2.2000 or higher

Insurance Apps Dependencies: Insurance Product Factory v4.3.0.

Upgrade from: N/A.

Install instructions: follow the standard procedure on how to import a digital asset. After installing the digital solution, run the SQL script from **Insurance Product Factory Import** folder.

After installment configurations: N/A.

Insurance Product Factory Import Package v4.2.0

Digital asset description: Data import configuration package for insurance products, insurance risks and insurance types. Contains demo insurance products and versioning settings.

Platform version compatibility: High Productivity Fintech Infrastructure 21.2.2

Dependencies: SySDigitalSolutionPackages v21.2.2000 or higher

Insurance Apps Dependencies: Insurance Product Factory Import Formulas v4.2.0

Upgrade from: N/A.

Install instructions: follow the standard procedure on how to import a digital asset.

After installment configurations: N/A.

Solution Configurations

The **Insurance Product Factory** solution enables you to create, adjust and maintain **Insurance Products**. Check the following pages to find out more about how this solution works:

Business Formulas Integration - for details about the integration with the **Business Formulas** engine.

Product Factory Endpoints - for details about what kind of API calls the solution allows you to make.

Cloning Insurance Products - for details about how the solution can replicate all product data into a new product.

Importing Insurance Products - for details about adding product data in the system by using import templates.

Configuration FAQs

- Insurance product configuration starts by adding a new record into FTOS_IP_InsuranceProduct entity, with general details about the product: name, code, settings influencing other flows' behavior on **Policy Admin** or **Billing** solutions.
- The coverages and the modules are saved in the same entity - that is the FTOS_IP_InsuranceProductItem entity, a coverage being always a parent of a module. A coverage is always linked to a product by **insuranceProductId** lookup; a module will not have the **insuranceProductId** lookup filled in, but instead it will have **itemParentId** lookup filled in with one of the existing coverages.
- The general risk nomenclature table is FTOS_IP_InsuranceRisk, but since a risk can have different configuration in the context of a certain product (like different limits), you must also use the FTOS_IP_CoveredRisk to create multiple "flavors" for a specific risk.

- A covered risk can be associated on multiple modules, and a module can have multiple covered risks. The FTOS_IP_InsuranceCoveredRisk entity is used to create the necessary links between them.
- In order to calculate the premium, formulas can be associated to each coverage of a product. The FTOS_IP_InsuranceProductItemFormula entity is used to link a coverage (FTOS_IP_InsuranceProductItem) with a formula (FTOS_CALC_ Formula). Pay attention to the fact that only one formula can be attached on a specific coverage of a product.
- On the other hand, underwriting rules are associated directly on the product, not on coverages, and FTOS_IP_InsuranceProductItemFormulaUnd entity stores the links between a product (FTOS_IP_InsuranceProduct) and a formula (FTOS_ CALC_Formula). When adding new records in FTOS_IP_ InsuranceProductItemFormulaUnd, a **context type attribute is required**, since underwriting rules can be divided in different categories, defined in FTOS_IP_ UnderwrittingContextType. Pay attention to the fact that only one formula for underwriting can be associated on a product, on a specific context.

HINT
The **Insurance Product Factory** maintains the underlying data in a consistent data model. Consequently, you have a reliable data model for your insurance product portfolio that you can reference when you build your digital journeys. This allows you to manage your product portfolio at will, without having to re-code your digital journeys every time an insurance product is added, updated, or retired.

Insurance Product Factory Endpoints

There are cases when you need to access the product data or **Insurance Product Factory** functionalities in a way alternative to the one available as a user journey, in **Innovation Studio**. The solution allows you to make API calls regarding different product-related aspects. For interacting with defined products, the following endpoints are available:

- FTOS_IP_PremiumAmountAPI - for getting prices on specified insurance product items.

- FTOS_IP_GetUWRulesResultAPI - for getting the underwriting decision for a specified insurance product item.
- FTOS_IP_GetProductFormulasStructuresAPI - for getting all the formulas attached to a specified insurance product.

Apart from the above endpoints, the following are all the other endpoints available with the Insurance Product Factory solution:

FTOS_IP_CALC_TestPremiumAmount

Script name: FTOS_IP_CALC_TestPremiumAmount

Description: This script is used during the **Test Calculation** step, to test the premium calculation formulas attached on insurance product coverages and return the result.

FTOS_IP_CALC_TestVerifyUnderwriting

Script name: FTOS_IP_CALC_TestVerifyUnderwriting

Description: This script is used during the **Test Calculation** step, to test the underwriting rules on an insurance product and return the result.

FTOS_IP_CloneIP

Script name: FTOS_IP_CloneIP

Description: This script is triggered when pressing the **Clone** button, on an insurance product. It gets all the configurations attached to a specified insurance product and uses them to create a new clone product.

FTOS_IP_InsuranceProductItemFormula_CreateDataMapping

Script name: FTOS_IP_InsuranceProductItemFormula_CreateDataMapping

Description: This script creates the data mapping for the premium calculation formulas/ rules attached on product items.

FTOS_IP_InsuranceProductItemFormulaUnd_CreateDataMapping

Script name: FTOS_IP_InsuranceProductItemFormulaUnd_CreateDataMapping

Description: This script creates the data mapping for the underwriting calculation formulas/ rules attached on product items.

FTOS_IP_ValidateIP

Script name: FTOS_IP_ValidateIP

Description: This script is used in the cloning process, to check if the provided product name and code are not already used on another product.

FTOS_VersioningHelper_Edit_FetchEntity

Script name: FTOS_VersioningHelper_Edit_FetchEntity

Description: This script handles server-side operations needed in the versioning process, when evaluating whether the versioning option should be available on a record or not.

> **NOTE**
> For more details about the endpoints description, request data parameters, response, error messages, and more, go to their respective pages, using the links listed above.

Get Formulas Structures API

Use this API to get the keys expected by the formulas attached on coverages (product items) or on product. The API will return information about all the formulas attached on the coverages of the products and all the structures for underwriting formulas attached on product level, for each context type. If necessary, in the API call you can specify a date and get the formula structure that was valid at that date, for the selected product.

Example

A user makes a call for the formula structure for an insurance product, based on the code of that product. The user wants the formula structure which was valid at a certain date - respectively, 2021-05-21.

```
let p = {
    "productCode": "BA_EMB"
    "validityDate": "2021-05-21"
}
ebs.callActionByNameAsync("FTOS_IP_
GetProductFormulasStructuresAPI", p)
    .then(function(e) {
        console.log(e)
    })
```

Request data parameters

Here is the list of data parameters included in the request:

Parameter	Description
productCode	The code of the **Insurance Product**.
validityDate	The reference date, prior to the current date, for getting **an earlier version** of the product formulas structure. This key is not mandatory. When it is not provided, the API calls the current version of the formula. Accepted formats are: yyyy-mm-dd, dd/mm/yyyy, dd-mm-yyyy, or dd.mm.yyyy.

Response

This is an example of a response:

```
{
  "isSuccess": true,
  "errorMessage": null,
  "errorCode": null,
  "result": [
    {
      "productCode": "BA_EMB",
      "premiumCalculationStructures": [
        {
          "code": "HHR0",
          "formulaVersion": 1.0,
          "formulaStructure": [
```

```
            {
              "key": "risks",
              "value": null,
              "masterType": "Collection",
              "subType": "Object",
              "objProps": "
{\"Risk\":6,\"Module\":6,\"Price\":2}"
            },
            {
              "key": "propertyInsuredAmount",
              "value": null,
              "masterType": "SimpleType",
              "subType": "Decimal",
              "objProps": null
            },
            {
              "key": "propertyConstructionYear",
              "value": null,
              "masterType": "SimpleType",
              "subType": "WholeNumber",
              "objProps": null
            }
          ]
        },
        {
          "code": "HA",
          "formulaVersion": 1.0,
          "formulaStructure": [
            {
              "key": "buildingInsuredAmount",
              "value": null,
              "masterType": "SimpleType",
              "subType": "Decimal",
              "objProps": null
            },
            {
              "key": "constructionYear",
              "value": null,
              "masterType": "SimpleType",
              "subType": "WholeNumber",
              "objProps": null
            }
          ]
        }
      ],
```

```
        "uwRulesStructures": [
          {
            "contextType": "Underwriting",
            "formulaVersion": 1.0,
            "formulaStructure": [
              {
                "key": "insuredAmount",
                "value": null,
                "masterType": "SimpleType",
                "subType": "Decimal",
                "objProps": null
              },
              {
                "key": "seismicZone",
                "value": null,
                "masterType": "SimpleType",
                "subType": "Text",
                "objProps": null
              }
            ]
          }
        ]
      }
    ]
}
```


Response description:

Key	Description
Error code	Error code.
Error message	Error message.
isSuccess	Marks whether the request was successful or not.
result	Array of objects containing details about the prices.
productCode	The code of the **Insurance Product**.
premiumCalculationStructures	Array with each premium formula input parameters attached to a product item.
uwRulesStructures	Array with each underwriting formula input parameters attached to a product .

Error Messages

The following are the error messages that can be encountered while calling the **GetFormulasStructuresAPI**:

Code	Text	Description
ERR.IP.50201	ERR.IP.50201 - Invalid validity date format! Please, use yyyy-mm-dd or dd/mm/yyyy or dd-mm-yyyy or dd.mm.yyyy!	Invalid date format for **validityDate** input parameter.
ERR.IP.50202	ERR.IP.50202 - Invalid validity date!	Invalid date for **validityDate** input parameter.
ERR.IP.50203	ERR.IP.50203 - Invalid key request!	Missing parameters in the request or wrong parameter name.
ERR.IP.50204	ERR.IP.50204 - No active product identified!	No active insurance product found.
ERR.IP.50205	ERR.IP.50205 - Error! The //code of product// insurance product was not approved at the requested date!	No active insurance product found at the date specified in **validityDate** input parameter.
ERR.IP.50206	ERR.IP.50206 - Underwriting context type is not defined!	The underwriting context type is not defined.

Endpoints

The **FTOS_IP_GetProductFormulasStructuresAPI** endpoint is responsible for getting the formulas input parameters structure for premium calculation for each product item and, also, for getting the formulas input parameters structure for the underwriting rules defined on product level.

Server Side Script Libraries

From the **FTOS_IP_InsuranceProductAPIs** library, the **GetFormulaInputParameters()** function is used for getting the input parameters structure for the premium and underwriting formulas. The **GetFormulaInputParameters()** function wraps the following functions, necessary for validating and returning the input parameters structure:

validateRequest()

This function validates the request fields.

Input parameters: `inputData` - The object containing the keys needed to call the endpoint.

Output parameters: An `object` containing the following **keys**, for describing the result of the validation:

- `isSuccess` - true/ false - The boolean that shows whether the validation is successful.
- `errorMessage` - A null value or an error message as described in the error messages list.
- `errorCode` - A null value or an error code as described in the error messages list.
- `result` = An empty array [] or an array with details about the input parameters formula structure, as described in the response description section.

getPremiumFormulaStructure()

This function gets the premium formulas input parameters attached on the specified product items - identified by the code provided in the request. The function also uses other helper functions, implemented in the same library, to get the product details, the list of items, and the formulas attached on items.

Input parameters: `inputData` - The object containing the keys needed to call the function.

Output parameters: An `array` containing objects with 2 keys:

- `productCode` - The code of the insurance product.
- `premiumCalculationStructures` - An array containing the following: **objects** for each product item, **objects** for the item code and the formula version number and **objects** for each input parameter.

getUWFormulaStructure()

This function gets the underwriting formulas input parameters attached on the specified product - identified by the code provided in the request. The function also uses other helper functions, implemented in the same library, to get the product details and the underwriting formulas attached on product.

Input parameters: `inputData` - The object containing the keys needed to call the function.

Output parameters: An `array` containing objects with 2 keys:

- `productCode` - The code of the insurance product.
- `uwRulesStructures` - An array containing the following: **object** for the requested product item, **object** for the context type formula and the formula version number and, also, an **object** for each input parameter.

getFormulaStructure()

This function concatenates the results of the **getPremiumFormulaStructure** and **getUWFormulaStructure** functions into an array. This function is called inside the endpoint only if the request passes the validation - namely if “isSuccess” key from the response of **validateRequest** function is true.

Input parameters: `inputData` - The object containing the keys needed to call the endpoint **FTOS_IP_ GetProductFormulasStructuresAPI**.

Output parameters: An `array` containing objects with 2 keys:

- `productCode` - An object from **getPremiumFormulaStructure** output parameters.
- `premiumCalculationStructures` - An object from **getPremiumFormulaStructure** output parameters.

Get Premium Amount API

Use this API to calculate the premium for each coverage (product item) of an insurance product. For each product item, the input keys expected by formula engine are required. In order to get the premium calculation, the **FTOS_IP_PremiumAmountAPI** endpoint calls the formulas attached on each item included in the request and identified by its code. If necessary, in the API call you can specify a date and get the price that was valid at that date, for your product item.

Example

A user makes a call for the price (premium amount) for a product item configured on a property insurance product. The user wants the premium amount formula which was valid at a certain date - respectively, 2021-05-21.

```
var  p = {
   "insuranceTypeName": "Home",
   "productCode": "BA_EMB",
   "validityDate": "2021-05-21",
   "insuranceProductItemDetails": [{
       "code": "HHR0",
       "calculationDetails": {
           "risks": [{
               "Risk": "Earthquake",
               "Module": "M1",
               "Price": 0
           }, {
               "Risk": "Land Slide",
               "Module": "M1",
               "Price": 0
           }, {
               "Risk": "Floods",
               "Module": "M1",
               "Price": 0
           }, {
               "Risk": "Windstorm",
               "Module": "M1",
               "Price": 0
           }],
           "propertyInsuredAmount": 100000,
           "propertyConstructionYear": 1980
```

```
            }
        }, {
            "code": "HA",
            "calculationDetails": {
                "buildingInsuredAmount": 23.45,
                "constructionYear": 100
            }
        }]
};

ebs.callActionByNameAsync("FTOS_IP_PremiumAmountAPI",p)
.then(function(e){
    console.log(e)
})
```

Request Data Parameters

Here is the list of data parameters included in the request:

Parameter	Description
insuranceTypeName	The name of an **Insurance Type** configured in the system and stored on the **Insurance Type** entity.
productCode	The code of the **Insurance Product**.
validityDate	The reference date, prior to the current date, for calling **an earlier version** of the formula. This key is not mandatory. When it is not provided, the API calls the current version of formula. Accepted formats are: yyyy-mm-dd, dd/mm/yyyy, dd-mm-yyyy, or dd.mm.yyyy.
insuranceProductItemDetails	Array with details to identify and run the formulas.
code	The code of the **Item**.
calculationDetails	Object containing the keys needed to run the formula. The object structure can be different for each item, based on the formula configuration.

Response

This is an example of a response:

```
 {
    "isSuccess": true,
    "errorMessage": null,
    "errorCode": null,
    "result": [
        {
            "insuranceProductItemFormula":
"EmbededBankAssurancePremiumAmountFormula",
            "formulaResult": {
                "risks": [
                    {
                        "Risk": "Earthquake",
                        "Module": "M1",
                        "Price": 0
                    },
                    {
                        "Risk": "Land Slide",
                        "Module": "M1",
                        "Price": 0
                    },
                    {
                        "Risk": "Floods",
                        "Module": "M1",
                        "Price": 0
                    },
                    {
                        "Risk": "Windstorm",
                        "Module": "M1",
                        "Price": 0
                    }
                ],
                "propertyInsuredAmount": 100000,
                "propertyConstructionYear": 1980,
                "risksCalculatedPrices": [
                    {
                        "Risk": "Earthquake",
                        "Module": "M1",
                        "Price": 0.0054
                    },
                    {
                        "Risk": "Land Slide",
                        "Module": "M1",
                        "Price": 0.001
                    },
                    {
```

```
                        "Risk": "Floods",
                        "Module": "M1",
                        "Price": 0.001
                    },
                    {
                        "Risk": "Windstorm",
                        "Module": "M1",
                        "Price": 0.013
                    }
                ],
                "baseQuotation": 0.0204,
                "premiumAmount": 20.4
            }
        },
        {
            "insuranceProductItemFormula":
"HomeAssistance",
            "formulaResult": {
                "buildingInsuredAmount": 23.45,
                "constructionYear": 100,
                "coefHomeAssistancePrice": 0.001,
                "premiumAmount": 0.0002345
            }
        }
    ]
}
```


Response description:

Key	Description
Error code	Error code.
Error message	Error message.
isSuccess	Marks whether the request was successful or not.
result	Array of objects containing details about the prices.
insuranceProductItemFormula	Name of a record from the **Insurance Product Item Formula** entity, containing the **Formula** attached on a **Product Item**.
formulaResult	The result returned by the **Formula**.
code	The insurance product item code.

Error Messages

The following are the error messages that can be encountered while calling the **Get Premium Amount API**:

Code	Text	Description
ERR.IP.50101	ERR.IP.50101 - Invalid validity date format! Please, use yyyy-mm-dd or dd/mm/yyyy or dd-mm-yyyy or dd.mm.yyyy!	Invalid date format for **validityDate** input parameter.
ERR.IP.50102	ERR.IP.50102 - Invalid validity date!	Invalid date for **validityDate** input parameter.
ERR.IP.50103	ERR.IP.50103 - Invalid request!	Missing parameters in the request.
ERR.IP.50104	ERR.IP.50104 - No active product identified!	No active insurance product found.
ERR.IP.50105	ERR.IP.50105 - Error! The //code of product// insurance product was not approved at the requested date!	No active insurance product found, for the date specified in the **validityDate** input parameter.
ERR.IP.50106	ERR.IP.50106 - Product doesn't have any product items configured!	The insurance product identified has no insurance items.
ERR.IP.50107	ERR.IP.50107 - Invalid insurance item code.	Invalid product item code.

Endpoints

The **FTOS_IP_PremiumAmountAPI** endpoint calculates the premium amount for the items included in a product configuration. The endpoint runs the formulas assigned to each item, simulating the **Test Scenario** functionality available on product level, in **Innovation Studio**.

Server Side Script Libraries

From the **FTOS_IP_InsuranceProductAPIs** library, the following functions are used for getting the price for the items:

validateRequest()

This function validates the request fields.

Input parameters: `inputData` - The object containing the keys needed to call the endpoint.

Output parameters: An `object` containing the following **keys** to describe the result of the validation:

- `isSuccess` - true/ false - The boolean that shows whether the validation is successful.
- `errorMessage` - A null value or an error message as described in the error messages list.
- `errorCode` - A null value or an error code as described in the error messages list.
- `result` = An empty array [] or an array with details about the price, as described in the response description section.

calculatePremium()

This function runs the formulas attached to the specified items - identified by the code provided in the request. The function uses other helper functions, implemented in the same library, to get the product details on a specific date, the list of items, and the formulas attached to the specified items. The `calculatePremium()` function is called inside the endpoint only if the **isSuccess** key from the response of the `validateRequest` function is true.

Input parameters: `inputData` - The object containing the keys needed to call the endpoint.

Output parameters: An `array` containing objects with 2 keys:

- `insuranceProductItemFormula` - The name of the formula used to calculate the price for the item.
- `formulaResult` - The actual result returned when calling the formula.

Get UW Rules Result API

Use this API to run underwriting formulas, based on the underwriting rules (UW Rules) defined at product level. In order to use the API, the request needs to include all the keys expected by formula engine, for each context type. If necessary, in the API call you can specify a date and get the underwriting decision that was valid at that date, for your product.

Example

A user makes a call for the underwriting decision result for an insurance product item, based on the code of the insurance product. The user wants the decision which was valid at a certain date - respectively, 2021-05-21.

```
let p = {
    "productCode": "HH01",
    "validityDate": "2021-05-21",
    "uwRules": [{
        "contextType": "Underwriting",
        "details": {
            "resistanceStructure": "Metal",
            "usageType": "Main residence",
            "constructionYear": 2000,
            "coverage": "Content",
            "insuredAmount": 200000
        }
    }]
};

ebs.callActionByNameAsync("FTOS_IP_GetUWRulesResultAPI", p).then(function(e) {
    console.log(e)
})
```

Request data parameters

Here is the list of data parameters included in the request:

Parameter	Description
productCode	The code of the **Insurance Product**.

Parameter	Description
validityDate	The reference date, prior to the current date, for getting **an earlier version** of the product formulas structure. This key is not mandatory. When it is not provided, the API calls the current version of the formula. Accepted formats are: yyyy-mm-dd, dd/mm/yyyy, dd-mm-yyyy, or dd.mm.yyyy.
uwRules	Array with the objects keys listed and described bellow. The array is used to identify and run the underwriting formulas for the specified **Item**.
contextType	The context type object key needed to run the UW formula.
details	An object containing the keys needed to run the UW formula. The object structure can be different for each product, based on the formula configuration.

Response

This is an example of a response:

```
{
  "isSuccess": true,
  "errorMessage": null,
  "errorCode": null,
  "result": [
    {
      "contexType": "Underwriting",
      "decision": {
        "resistanceStructure": "Metal",
        "usageType": "Main residence",
        "constructionYear": 2000.0,
        "coverage": "Content",
        "insuredAmount": 200000.0,
        "uwBuildingSumInsured": "Passed",
        "uwConstructionYyear": "Passed",
        "uwResistanceStructure": "Passed",
        "uwUsageType": "Passed",
        "uwDecision": "Passed"
      }
    }
  ]
}
```

Response description:

Key	Description
Error code	Error code.
Error message	Error message.
isSuccess	Marks whether the request was successful or not.
result	Array of objects containing details about the UW decision results.
contextType	The context type defined on UW formula (underwriting).
decision	The result returned after running the UW formula.

Error Messages

The following are the error messages that can be encountered while calling the **Get UW Rules API**:

Code	Text	Description
ERR.IP.50301	ERR.IP.50201 - Invalid validityDate format! Please, use yyyy-mm-dd or dd/mm/yyyy or dd-mm-yyyy or dd.mm.yyyy!	Invalid date format for **validityDate** input parameter.
ERR.IP.50302	ERR.IP.50202 - Invalid validityDate!	Invalid date for **validityDate** input parameter.
ERR.IP.50303	ERR.IP.50303 - Invalid key request! + “inputKeyName”+ "!	Missing parameters in the request or wrong parameter name.
ERR.IP.50304	ERR.IP.50304 - No active product identified!	No active insurance product found.
ERR.IP.50305	ERR.IP.50305 - Error! The "code of product" insurance product was not approved at the requested date!"	No active insurance product factory found at the date specified in **validityDate** input parameter.
ERR.IP.50306	ERR.IP.50306 - Underwriting context type is not defined!	The underwriting context type is not defined on the requested product.
ERR.IP.50307	ERR.IP.50307 - Product doesn't have any UW formula configured.	Product doesn't have any UW formula configured.
ERR.IP.50308	ERR.IP.50308 - Invalid context type value: + “contextTypeInputValue”.	The key for underwriting context type has a wrong value.

Code	Text	Description
ERR.IP.50309	ERR.IP.50309 - Formula definition includes other input values. Check the definition: + “invalidCodes”.	Formula input parameters names are wrong.

Endpoints

The **FTOS_IP_GetUWRulesResultAPI** endpoint is responsible for running the underwriting formula defined on product level, and retuning the underwriting decision results. The endpoint runs the formulas assigned to the product, simulating the **Test Scenario** functionality available on product level, in **Innovation Studio**.

Server Side Script Libraries

From the **FTOS_IP_InsuranceProductAPIs** library, the **GetUWRulesResult()** function is used for returning the underwriting decision results. The **GetFormulaInputParameters()** function wraps the following functions, necessary for validating and returning the results:

validateRequest()

This function validates the request fields.

Input parameters: `inputData` - The object containing the keys needed to call the endpoint.

Output parameters: An `object` containing the following **keys** to describe the result of the validation:

- `isSuccess` - true/ false - The boolean that shows whether the validation is successful.
- `errorMessage` - A null value or an error message as described in the error messages list.
- `errorCode` - A null value or an error code as described in the error messages list.

- `result` = An empty array [] or an array with details about the UW formula input parameters , as described in the response description section.

getUWRulesResult()

This function executes the following:

- Runs the underwriting formulas attached to the specified product - identified by the code provided in the request.
- Uses other helper functions, implemented in the same library, to get the product details.
- Calls the **testUnderwritting** function from **FTOS.IP_CALC.FormulaEngineHelper** library.
- Returns the underwriting decision results.

The `getUWRulesResult()` function is called inside the endpoint only if the **isSuccess** key from the response of the `validateRequest` function is true.

Input parameters: `inputData` - The object containing the keys needed to call the function.

Output parameters: An `array` containing objects with 2 keys:

- `contextType` - The underwriting context type used for the selected product.
- `decision` - An object containing the keys and the values from the input object (details about the UW rules) and also the corresponding keys and values for decision results.

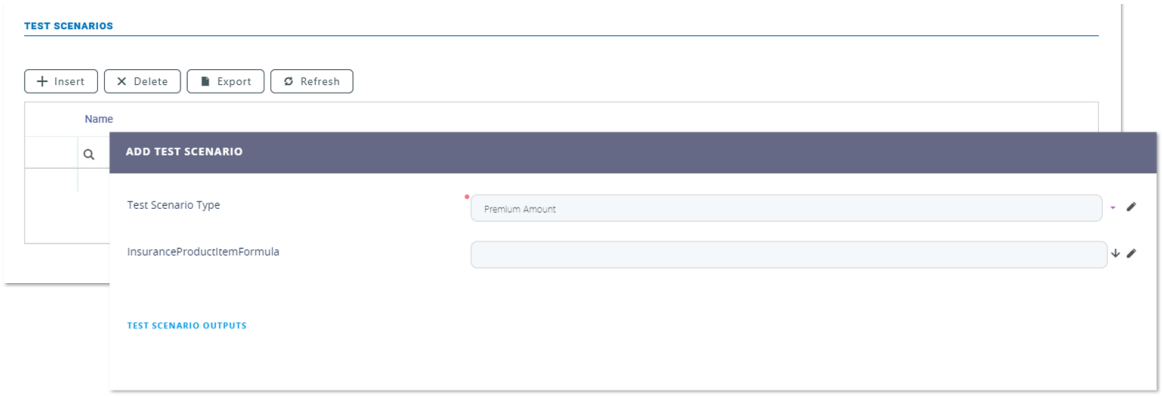
Business Formulas Integration

The **Insurance Product Factory** module integrates smoothly with the **FintechOS** Business Formulas solution which enables you to create your own formulas for insurance. **Formulas** can support various business needs and they are a mandatory prerequisite for making your **Insurance Product Factory** work at its full capacity.

You input your financial and non-financial data and arguments into your **Formulas** and, consequently, they help you with risk scoring - for example, to determine whether a property is eligible for insurance or not. In addition, **Formulas** generate accurate desired outputs - such as premium calculations split per insured items and handle various calculation scenarios - for example, for when premiums are configured differently for individual or group purchases.

Use **Business Formulas** to create all the necessary formulas for handling the premium calculations for the specified **Insurance Product Items** and then use the Premium Amount tab (inside a chosen product) to attach them to the product items. Similarly, use **Business Formulas** to create all the necessary formulas for handling the risk scoring for your **Insurance Product** and then use the Underwriting tab (inside a chosen product) to attach them to the product.

The **Insurance Product Factory** solution also allows you to test your formulas based on different scenarios that you can log onto the **Test Scenarios** page. Using scenarios to test your formulas helps with fine-tuning and improving the general performance of your **Insurance Product**.



NOTE
Only one formula can be attached on a specific coverage of a product.

Demo Formulas

The following formulas are included in the **Import Formulas** digital asset, for demo purposes:

Formula 1 - Bankassurance_UW_Formula_Final

Formula Steps

The following are the steps used for risk scoring:

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Exection Order
ruleInsuredAmount	FALSE	Simple Type	Text	Normal	1
ruleSeismicZone	FALSE	Simple Type	Text	Normal	2
uwDecision	FALSE	Simple Type	Text	Normal	3

Formula Input

The following are the arguments used as input for the **Bankassurance_UW_Formula_Final** calculations: `insuredAmount` and `seismicZone`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
BA_UW_ SeismicZone	Bankassurance UW Seismic Zone	SeismicZone	BA_UW_ SeismicZone
BA_UW_ InsuredAmount	Bankassurance_ UW_ InsuredAmount	InsuredAmount	BA_UW_ InsuredAmount
BA_UW_ Surrender	Bankassurance UW Surrender	hasSurrender	BA_UW_ Surrender

Formula 2 - EmbededBankAssurancePremiumAmountFormula

Formula Steps

The following are the steps used for premium calculation, with this formula:

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Exection Order
risksCalculatedPrices	FALSE	Collection	Whole Number	Iteration	1
baseQuotation	FALSE	Simple Type	Decimal	Normal	2
premiumAmount	FALSE	Simple Type	Decimal	Normal	3

Formula Input

The following are the arguments used as input for the formula calculations: `propertyConstructionYear`, `propertyInsuredAmount`, and `risks`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
BankAssurance_ RisksPrices	BankAssurance Risks Prices	ConstructionYear; RiskName	BankAssurance_ RisksPrices

Formula 3 - HomeServiceBankAssurancePremiumAmountFormula

Formula Steps

The following are the steps used for premium calculation, with this formula:

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Exection Order
coefHomeAssistancePrice	FALSE	Simple Type	Decimal	Normal	1
premiumAmount	FALSE	Simple Type	Decimal	Normal	2

Formula Input

The following are the arguments used as input for the formula calculations: `buildingInsuredAmount` and `constructionYear`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
BA_HomeService_Prices	Bankassurance Home Service Prices	ConstructionYear	BA_HomeService_Prices

Formula 4 - PA_Final_Premium

Formula Steps

The following are the steps used for premium calculation:

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Execution Order
BaseRate	FALSE	Simple Type	Decimal	Normal	1
CoefFrequency	FALSE	Simple Type	Decimal	Normal	2
CoefAge	FALSE	Simple Type	Decimal	Normal	3
CoefAmountInsuredItem	FALSE	Simple Type	Decimal	Normal	4
ItemPremiumAmount	FALSE	Simple Type	Decimal	Normal	5

Formula Input

The following are the arguments used as input for the **PA_Final_Premium** calculations: `age`, `coverage`, `frequency` and `sumInsured`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
PA_Age	PA_Age	`Age`	Age
PA_AmountInsuredItem	Amount Insured Item	`AmountInsured`; `Coverage`	AmountInsuredItem
PA_Base_Rate	PA_Base_Rate	`Coverage`	PA_Base_Rate
PA_Frequency	PA_Frequency	`Frequency`	PA_Frequency

Formula 5 - PropertyFormula

Formula Steps

The following are the steps used for premium calculation.

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Exection Order
BaseRate	FALSE	Simple Type	Decimal	Normal	1
CoefFrequency	FALSE	Simple Type	Decimal	Normal	2
CoefAge	FALSE	Simple Type	Decimal	Normal	3
CoefAmountInsuredItem	FALSE	Simple Type	Decimal	Normal	4
ItemPremiumAmount	FALSE	Simple Type	Decimal	Normal	5

Formula Input

The following are the arguments used as input for the **PA_Final_Premium** calculations: `age`, `coverage`, `frequency` and `sumInsured`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
HH_BaseRate	HH_BaseRate	Coverage	HH_BaseRate
HH_BuidingSumInsured	HH_BuidingSumInsured	Coverage; InsuredAmount	HH_BuidingSumInsured
HH_BuildingType	HH_BuildingType	BuildingType; Coverage	HH_BuildingType
HH_Construction_Year	Construction Year	Construction Year; Coverage	HH_Construction_Year

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
HH_Frequency	HH_Frequency	Frequency	Frequency
HH_ ResistanceStructure	HH_ ResistanceStructure	Coverage; Resistance Structure	HH_ ResistanceStructure
HH_UsageType	Usage Type	Coverage; Usage Type	HH_UsageType

Formula 6 - PropertyFormula_UW

Formula Steps

The following are the steps used for premium calculation.

Name	Exclude From Mapping	Master Type	SubType	CalculationType	Exection Order
BaseRate	FALSE	Simple Type	Decimal	Normal	1
CoefFrequency	FALSE	Simple Type	Decimal	Normal	2
CoefAge	FALSE	Simple Type	Decimal	Normal	3
CoefAmountInsuredItem	FALSE	Simple Type	Decimal	Normal	4
ItemPremiumAmount	FALSE	Simple Type	Decimal	Normal	5

Formula Input

The following are the arguments used as input for the **PA_Final_Premium** calculations: `age`, `coverage`, `frequency` and `sumInsured`.

Data sets

The following data sets are used for calculations:

Data Set Name	Data Set Display Name	Data Set Discriminants	Data Set Values
HH_UW_BuildingSumInsured	HH_UW_BuildingSumInsured	Coverage; InsuredAmount	HH_UW_BuildingSumInsured
HH_UW_ConstructionYear	HH_UW_ConstructionYear	ConstructionYear	HH_UW_ConstructionYear
HH_UW_ResistanceStructure	HH_UW_ResistanceStructure	ResistanceStructure	HH_UW_ResistanceStructure
HH_UW_UsageType	HH_UW_UsageType	UsageType	HH_UW_UsageType

HINT
For more details about configuring formulas, see the Business Formulas documentation and this Risk Scoring tutorial.

Cloning Insurance Products

This functionality allows the user to generate a new insurance product based on an existing one. The cloned product captures all the available information from the original product. The user must select an existing product and press the **Clone** button available on the product view. This action triggers the opening of a specific pop-up form where the user can insert a new name and a new code for the new product. Upon saving, the clone product is generated, in **Draft** status.

Endpoints

The following endpoints are used to perform the cloning process:

FTOS_IP_ValidateIP

This endpoint receives an object as a parameter. The object contains the product name and code inserted in the cloning pop-up form, connected to the original product. The endpoint takes those values and validates them, to make sure there is no other product with the same name and code.

FTOS_IP_CloneIP

This endpoint receives an object as a parameter. The object contains the new product name and code, obtained from the original product form. The script linked to this endpoint takes these values and inserts a cloned product in the system. The clone product contains the same data, similar to the original one.

Server Side Script Libraries

The **FTOS_IP_InsuranceProduct_Clone** library is used to clone insurance products. From this library, the following functions are used:

GetData

This object wraps smaller functions that fetch data from different entities.

getProductInfo

This function fetches the **Name** and the **Code** of an insurance product based on a given name or code.

Input parameters:

- `entityName` - The name of the entity on which the fetch will be made.
- `ipName` - The name of the new insurance product.
- `ipCode` - The code of the new insurance product.

Output parameters: A list of insurance products, found by name or code.

getEntityId

This function fetches the **Id** of an entity based on a given name from the entity metadata.

Input parameters: `entityName` - The name of the entity needed.

Output parameters: The **Id** of the entity, found by the fetch.

checkEntityInVersionSettings

From the **Version Settings** entity, this function fetches all the records related to the entity that is about to be cloned.

Input parameters: `selectedEntity` - The Id of the entity from which the record is about to be cloned.

Output parameters: A list of all the related records.

getRelatedEntities

From the **Version Settings Item** entity, this function fetches all the items related to the entity that is about to be cloned.

> **NOTE**
> Parts of the product are stored in different entities, for more details see the Importing Insurance Products page.

Input parameters: `versionSettingsid` - The Id of the entity that is about to be cloned.

Output parameters: A list of all the related items.

getAlias

This function subtracts the first letter from an entity name to use it as an alias.

Input parameters: `entityName` - The name of the entity.

Output parameters: The **first letter** from the entity name.

getEntityData

This function fetches the data from an entity. The fetch uses the entity name received as input and it searches according to the conditions written by two other parameters: the attribute name and the value it should have.

Input parameters:

- `entityName` - The name of the entity on which the fetch is performed.
- `conditionAttrName` - The name of the attribute found inside the condition.
- `conditionRecord` - The value of the attribute found inside the condition.

Output parameters: A list of all the records returned by the fetch.

Validations

This object wraps smaller functions that validate different sets of data.

isNullOrEmpty

This function checks if the value received through a parameter is valid. If the value is valid, the function returns true. Otherwise, it returns false.

Input parameters: `value` - The value which is about to be validated.

Output parameters: **Boolean** value (true/ false).

validateNameCode

From the client side, this function receives the values inserted into the fields of the cloning pop-up - namely, the values for **Name** and **Insurance Product Code**. If the values come in as being empty or invalid, an exception will be thrown to inform the user. The same thing happens if the values are already used by the system (an existing insurance product has the same name or the product code).

Input parameters:

- `entityName` - The name of the entity on which the fetch will be made.
- `ipName` - The name of the new insurance product.
- `ipCode` - The code of the new insurance product.

Output parameters: If no exception is thrown, the function quits and the cloning process moves on.

ProductClone

This object wraps functions that perform various steps of the cloning process.

cloneRelatedEntities

This function receives a list of **Version Setting Items** as a parameter and loops through it for cloning each one of the items into the new product.

Input parameters:

- `versionSettingsItems` - The list of the version setting items.
- `productId` - The Id of the product which is about to be cloned.

- `newCloneId` - The Id of the new product that receives all the data from the original one.

Output parameters: N/A.

insertClonedProduct

This function inserts into an entity an object with data, both received as a parameter, after some validations are being made on a few attribute names.

Input parameters:

`entityName` - The name of the entity on which the insert will be made.

`entityData` - The object containing all the data from the original product.

Output parameters: Returns an Id of the newly inserted record.

cloneRelatedRecord

This function is similar with the function above. The function inserts into an entity an object with data, both received as a parameter, after some validations are being made on a few attribute names.

Input parameters:

- `entityName` - The name of the entity on which the insert is made.
- `entityData` - The object containing all the data from the original product.
- `recordId` - The Id of the product which is about to be cloned.
- `relatedAttr` - The name of an attribute that get as value the `recordId` parameter.

Output parameters: Returns an id of the newly inserted record

getChildEntities

This function the function firstly fetches from the entity metadata a record based on an name received through the parameter. After that, another fetch is made on the **Version Settings Item** entity using the Id obtained from the fetch before as a condition for the `parentVersionedEntity` attribute.

Input parameters: `entityName` - The name of the entity on which the fetch is made.

Output parameters: Returns a list of all the records found by the second fetch.

getPrimaryKeyValue

This function returns the value of a record's primary key from an entity with a specified name.

Input parameters:

- `entityName` - The name of the entity from which the primary key is needed.
- `record` - The record from the entity.

Output parameters: The **primary key** value.

clone

This function clones all the related entities of a product along with their "children" entities.

> **NOTE**
> Parts of the product are stored in different entities, for more details see the Importing Insurance Products page.

Input parameters:

- `entityName` - The name of the entity on which the fetch is made.
- `recordToBeClonedId` - The Id of the record which is about to be cloned.
- `recordToBeClonedPrimaryAttribute` - The Id of the record's primary attribute.
- `clonedRecordId` - The Id of the new record.

Output parameters: N/A

cloneProductData

This function calls all the other functions (detailed above) for validating, inserting and updating all the data for the product to be cloned.

Input parameters:

- `entityName` - The name of the entity on which the fetches are made.
- `productId` - The Id of the product that is about to be cloned.
- `ipName` - The name of the new insurance product.
- `ipCode` - The code of the new insurance product.

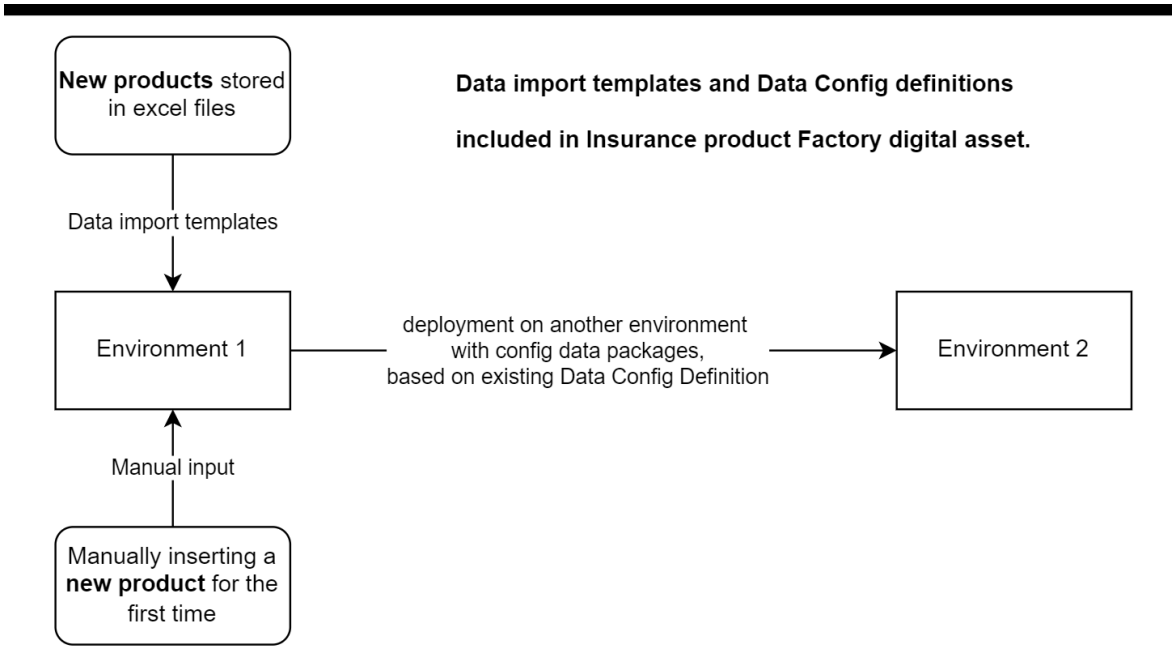
Output parameters: N/A

Importing Insurance Products

The **Data Import Templates** of the **Product Factory** solution allow you to add multiple types of product data on a **FintechOS** destination environment. The most important use case for this functionality is a fast bulk import into the system of numerous new insurance products - with all imported products being registered in **Draft** status. But this is not the only use case. For example, using the **Data Import Templates** provided by the **FintechOS High Productivity Fintech Infrastructure**, a user may want to import only some very specific data about certain product components. Consequently, templates are different depending on what kind of data needs to be added.

> **NOTE**
> A newly imported product is registered in **Draft** status, meaning it is editable. After import, use **Innovation Studio** to complete your product configurations and also, to attach premium calculation and underwriting formulas to it. Once finished, approve your product in order for it to go live.

The following is an example of use cases for importing insurance products:



See bellow the **Data Config Definitions** and **Data Import Templates** available for the **Insurance Product Factory** solution.

Data Import Templates

Use Data Import Templates to import a product for the first time on a destination environment. However, for deploying changes on existing products, use Data Config Definition, as it has the versioning mechanism embedded.

> **HINT**
> For downloading a specific example template file listed below, click the corresponding link. For downloading all the files, click this DataConfigImport Archive - which includes also the **DataConfigImport.xml** file.

The following are the **Data Import Templates** available for the **Insurance Product Factory** solution:

FTOS_IP_InsuranceType - This template will import a new **Insurance Type**.

FTOS_IP_InsuranceProduct - This template will import a new **Insurance Product**.

FTOS_IP_InsuranceItemType - This template will import a new **Insurance Item Type**.

FTOS_IP_InsuranceProductItem - This template will import a new **Coverage** on an **Insurance Product**.

FTOS_IP_InsuranceProductItemModule - This template will import a new **Module** on a **Coverage**.

FTOS_IP_InsuranceRisk - This template will import a new **Insurance Risk**.

FTOS_IP_CoveredRisk - This template will import a new **Covered Risk**.

FTOS_IP_InsuranceCoveredRiskNew - This template will import a new **Insurance Covered Risk** on a **Module**.

> **IMPORTANT!**
> After importing the files, additional checks or changes are need on product level, as not all the attributes from the entities are available in the excel files. Missing information needs to be filled in manually before approving the product. Also,

premium calculation formulas and underwriting formulas should be manually attached on the product.

Data Config Definition

The **FintechOSData Config Definition** functionality allows the deployment of products to a destination environment, using the versioning settings. The same versioning rules running for the manual versioning process will run when importing the data with Data Config Definition as well.

When your Product is in **Active** status, changing it may be done only by adding a **new version**. When you choose to import the product changes in a **Data Config Definition** file, this functionality has the versioning mechanism embedded. This way you can be sure that your changes are deployed to the specified product.

IMPORTANT!
The **Data Config Definitions** needed to export a product are available in the **Insurance Insurance Product Factory Import digital asset**. Examples are also attached on the **Data Import** files step of the digital asset. Not all the entities that are part of a product definition are included in the Product's **Data Config Definition**. For some entities, the data needs to be deployed using **Data Import Templates**.

The following deployment options are available:

FTOS_IP_InsuranceType

Use data import template to deploy the records.

FTOS_IP_InsuranceItemType

Use data import template to deploy the records.

FTOS_IP_InsuranceRisk

Data config definition available, exporting data from the **Insurance Risk** entity.

FTOS_IP_UnderwrittingContextType

Use data import templates to deploy the records.

FTOS_IP_InsuranceProduct

- Data config definition available, exporting data from the Insurance Product, Insurance Product Item (coverages), and Insurance Product Item (Modules) entities.
- **is Versionable flag enabled**
- **Premium Amount Formulas** and **Underwriting Formulas** will be exported as well. Make sure the **Formulas** are already available on the destination environment when you deploy the product.
-

> **NOTE**
>
> The Insurance Product Factory **Import Formulas Digital Asset** contains **Formulas** used by the demo **Product** included in the Insurance Product Factory **Import Digital Asset**.

-

> **IMPORTANT!**
>
> Following import, links with **Insurance Type**, **Insurance Item Type**, and **Underwriting Context Type** entity records will be made, so the needed records from these entities **must be imported first!**

FTOS_IP_CoveredRisk

Data config definition available, exporting records from **Covered Risk** entity and creating a link to **Insurance Risk** records, previously imported.

FTOS_IP_InsuranceCoveredRisk

Use data import template to deploy the records.

Below is an example of the **Order Index** for importing **Templates** and **Definitions**:

Data Import Template	Data Config Definition	Type	Order Index	File
FTOS_IP_InsuranceType		Based On Data Import Templates	1	Download
FTOS_IP_InsuranceItemType		Based On Data Import Templates	2	Download
	FTOS_IP_InsuranceRisk	Based On Data Config Definition	3	
FTOS_IP_UnderwrittingContextType		Based On Data Import Templates	4	Download
	FTOS_IP_InsuranceProduct	Based On Data Config Definition	5	
	FTOS_IP_CoveredRisk	Based On Data Config Definition	6	
FTOS_IP_InsuranceCoveredRiskNew		Based On Data Import Templates	7	Download

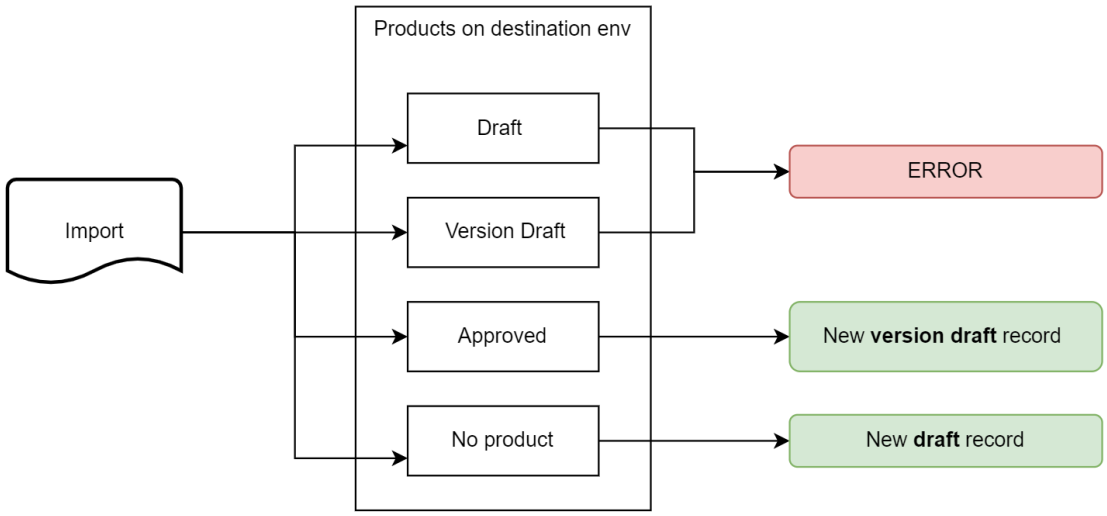
Troubleshooting Aspects

IMPORTANT!
Since the definition used to migrate records from the **Insurance Product** entity has the **is versionable** flag enabled, and since the versioning mechanism is available on products, **versioning rules are applying for each deployment**.

- If, on the destination environment, the **Product is NOT available** (identified by the unique constraint set on **Insurance Product** entity), then a new product with **Draft** status is added.
- If, on the destination environment, the **Product is already available** (identified by the unique constraint set on **Insurance Product** entity) and its status is **Draft**, then the import is not possible, and an error is triggered.

- If, on the destination environment, the **Product is already available** (identified by the unique constraint set on **Insurance Product** entity), and its status is **Approved** (with **NO Version Draft** records), then a new Product version (with **Version Draft** status) is created.
- If, on the destination environment, the **Product is already available** (identified by the unique constraint set on **Insurance Product** entity), and its status is **Approved** (with **Version Draft** records), then the import is not possible, and an error is triggered.

Below is an example of results obtained by importing between environments:



Server Side Scripts

FTOS_IP_InsuranceProduct_BeforeCreate

The script checks if the **Insurance Type** attribute is filled in when adding a new product. It runs every time a new product is created, either manually or using the import capabilities.

FTOS_IP_InsuranceProduct_BeforeUpdate

The script checks if the **Insurance Type** attribute is filled in when updating a new product.

FTOS_IP_InsuranceProductItem_BeforeCreate

The script stops the user to insert new **Coverages** or **Modules** on **Products** that **are not** in **Draft** or **Version Draft** status. It also ensures the coverages and modules are imported in the right place - that is the product and the product version indicated by the user. Inside this script, the following functions are used:

setCorrectProductId

This function ensures a **Coverage** is imported in the correct **Product** - in status **Draft** or **Version Draft**. When the user attempts to import a coverage on a product that is in other statuses (Approved, Closed, etc.), the following **error message** is thrown: “You can add a coverage only on draft or version draft products!”.

Input parameters: `context` - The context object received.

Output parameters: N/A

setCorrectCoverageId

This function ensures a **Module** is imported in the correct **Coverage**, for a product in status **Draft** or **Version Draft**. When the user attempts to import a module on a product that is in other statuses (Approved, Closed, etc.), the following **error message** is thrown: “You can add a coverage only on draft or version draft products!”.

Input parameters: `context` - The context object received.

Output parameters: N/A

FTOS_IP_InsuranceCoveredRisk_Import

The script ensures the **Covered Risks** are imported in the right place - that is the product and the product version indicated by the user. Inside this script, the following functions are used:

setCorrectModuleId

This function ensures the **Insurance Covered Risk** is imported on the correct **Module**.

Input parameters: `context` - The context object received.

Output parameters: N/A

getModuleData

This function fetches the initial module's data.

Input parameters: `insuranceProductItmeId` - The Id of the insurance product item.

Output parameters: The initial module's data.

getFinalModule

This function fetches the correct module's data.

Input parameters:

- `insuranceProductName` - The name of the insurance product.
- `coverageName` - The name of the coverage.
- `moduleName` - The name of the module.

Output parameters: The correct module's data.

HINT
For more details, consult also the Data Import Templates the Data Config Definition, and the Configuration Data Package documentation.

Digital Assets

The following **Digital Assets** are included in the **Insurance Product Factory** solution:

Insurance Product Factory Data Model

SDK Code: **IP_DataModel 4.3.0**

Description: This asset contains the configuration items defining the data model of the **Insurance Product Factory** solution. It also contains the business workflow attached to **FTOS_INSP_VersioningWorkflow** entity.

Items Contained: Actions, Attributes, BW, Config Data Definitions, Data Imports, Entities, Entity BWs, Entity Extensions, Entity Forms, Entity Views, Optionset Attributes, Workflow, Workflow Library.

Insurance Product Factory Form Driven Flows

SDK Code: **IP_Forms 4.3.0**

Description: This asset contains the **Form Driven Flows** related to the journey taken by the user when creating or adjusting an insurance product, in . The asset also contains entity views created for the entities included in the Data Model digital asset.

Items Contained: Entity Forms, Entity Views, Workflows.

Insurance Product Factory Import

SDK Code: **IP_Import 4.2.0**

Description: This asset is optional. It contains the following **demo products**, configured for different insurance types:

- **Term Life** (TLF) - for Life and Health insurance;
- **Personal Accidents** (PA) - for Life and Health insurance;

- **Household Insurance** (HH01) - for Property insurance;
- **Embedded** (BA_EMB) - for Property insurance;
- **Pet Insurance** (PETINS) - for Pet insurance.

The asset also includes data import files to load the demo products

Items Contained: Data Import Templates, Data Config Definitions.

Insurance Product Factory Menu

SDK Code: **IP_Menu 4.3.0**

Description: This asset contains the **Insurance Product Factory** menu items, displayed in **Innovation Studio**.

Items Contained: Menu Items.

Insurance Product Factory SDK

SDK Code: **IP_SDK 4.3.0**

Description: This asset contains all the scripts that run when creating or interacting with the products.

Items Contained: Actions, Client Script Library, Workflows, Workflow Libraries.

Insurance Product Factory Import Formulas

SDK Code: **IP_Import_Formulas 4.2.0**

Description: This asset is optional. It contains all the formulas (and their related data sets), needed to run scoring and pricing calculations for the demo products.

Items Contained: Formulas, Formula Inputs, Data Sets.

HINT
To inspect the **Insurance Product Factory** Digital Assets, go to **Innovation Studio** >

main menu, on the left > Configuration Management > Digital Assets > **Digital Assets List** Page and double-click on the desired Digital Asset to view general data about the Asset, the configuration items, related digital solutions, data import files, dependencies, and the configuration items migration section.

Glossary

Please check the **FintechOS** Getting Started Glossary!