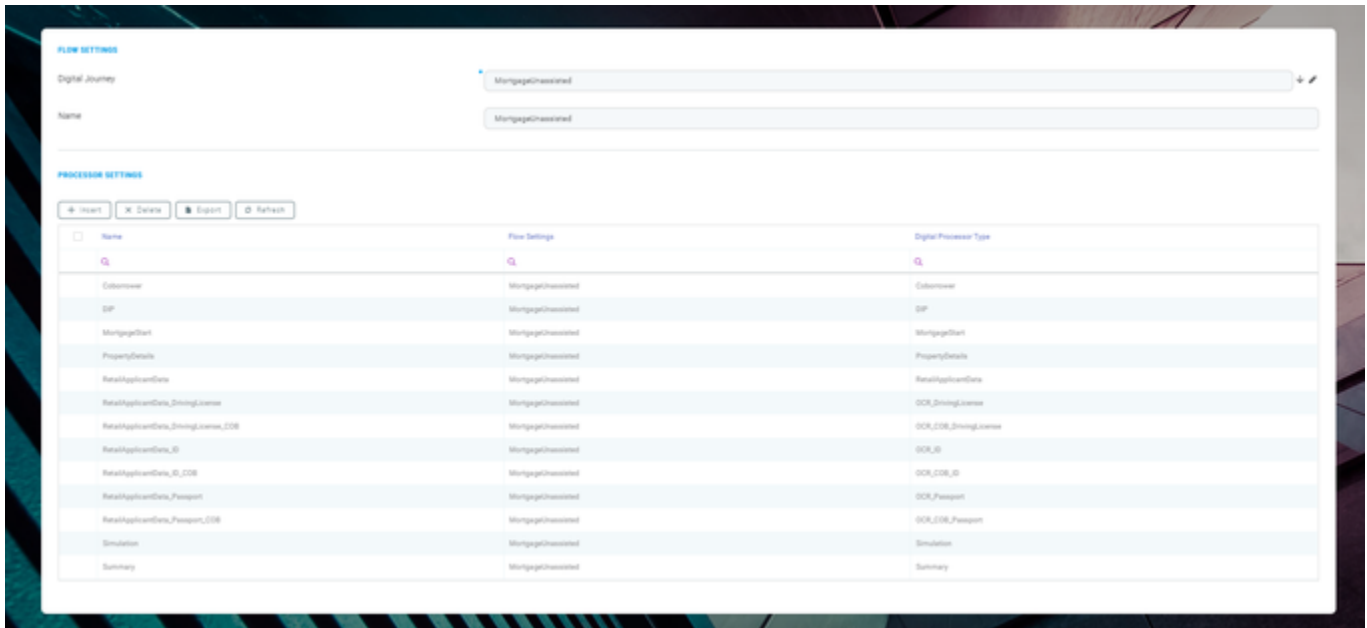


FlowSettings - why&where

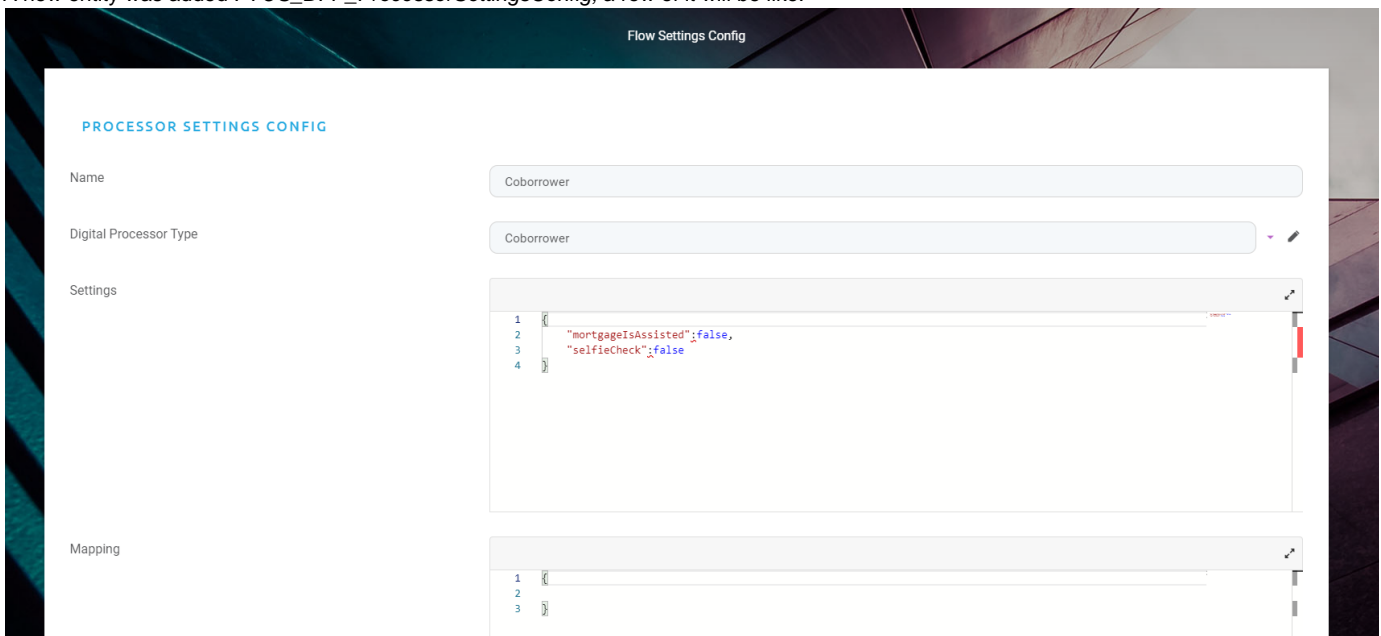


The screenshot shows the 'FLOW SETTINGS' interface. At the top, there's a 'Digital Journey' dropdown set to 'MortgageAssisted' and a 'Name' field also containing 'MortgageAssisted'. Below this is the 'PROCESSOR SETTINGS' section with buttons for '+ Insert', 'X Delete', 'E Export', and 'R Refresh'. A table lists various processor settings, all with 'MortgageAssisted' in the 'Flow Settings' column.

Name	Flow Settings	Digital Processor Type
Coborrower	MortgageAssisted	Coborrower
DP	MortgageAssisted	DP
MortgageStart	MortgageAssisted	MortgageStart
PropertyDetails	MortgageAssisted	PropertyDetails
RetailApplicationDate	MortgageAssisted	RetailApplicationDate
RetailApplicationDate_DrivingLicense	MortgageAssisted	OCR_DrivingLicense
RetailApplicationDate_DrivingLicense_ID	MortgageAssisted	OCR_ID_DrivingLicense
RetailApplicationDate_ID	MortgageAssisted	OCR_ID
RetailApplicationDate_ID_ID	MortgageAssisted	OCR_ID_ID
RetailApplicationDate_Passport	MortgageAssisted	OCR_Passport
RetailApplicationDate_Passport_ID	MortgageAssisted	OCR_ID_Passport
Simulation	MortgageAssisted	Simulation
Summary	MortgageAssisted	Summary

Flow settings will be used for a defined digital journey. On each step an endpoint will be called to fill in some virtual attributes with the digital journey id and the other settings like: mortgageIsAssisted.

A new entity was added *FTOS_DFP_ProcessorSettingsConfig*, a row of it will be like:



The screenshot shows the 'Flow Settings Config' form. It has four main sections: 'Name' (Coborrower), 'Digital Processor Type' (Coborrower), 'Settings' (a code editor with JSON), and 'Mapping' (a code editor with a simple structure).

PROCESSOR SETTINGS CONFIG

Name: Coborrower

Digital Processor Type: Coborrower

Settings:

```
1 {  
2   "mortgageIsAssisted": false,  
3   "selfieCheck": false  
4 }
```

Mapping:

```
1 {  
2     
3 }
```

Each time a new processor setting is added on the flow setting, the "Setting" field will be filled with the settings added on *FTOS_DFP_ProcessorSettingsConfig* for that specific digital processor type.

The scope of using this, is to have configs for each component, basically for each component will be a digital processor type and some settings. (You can set specific parameters that are using in the flow)

On the insert form of journey (*FTOS_BARET_MortgageB2C_Start*), the digital journey id is saved on *FTOS_BARET_Loan* entity. It's called endpoint "FTOS_GetDigitalJourneyNameById" to get the digital journey id using the name - **This is something temporary until a new version of application which will automatically save the digital journey id to the main entity.**

```
ebs.callActionByNameAsync("FTOS_GetDigitalJourneyNameById",
{digitalJourneyName: "MortgageUnassisted"})
    .then(function(r){
        if(r.IsSuccess){
            formData.model.digitalJourneyId = r.UIResult.Data.
digitalJourneyId;
        }
    })
```

On this specific scenario there are two main entities: FTOS_BARET_Loan & FTOS_BNKAP_RetailApplicantData. For these, some custom extensions with virtual attributes were added - both extensions are named FlowSettings:

ENTITY
FTOS_BARET_Loan

1 General 2 Virtual Attributes

BUSINESS ENTITY EXTENSION

Name

Extension Type

with the same custom attributes on both entities:

ENTITY
FTOS_BARET_Loan

1 General 2 Virtual Attributes

VIRTUAL ATTRIBUTES

+ Insert X Delete Export Refresh

Name	Display Name	Related Attribute	Updatable
flowSettings_digitalJourneyId	Journey		<input checked="" type="checkbox"/>
flowSettings_mortgageIsAssisted	Mortgage Is Assisted		<input checked="" type="checkbox"/>
flowSettings_selfieCheck	Selfie Check		<input checked="" type="checkbox"/>

Those attributes are added on UI as hidden.

On each form, the same call will be done:

```
var flowSettingCL = ebs.importClientScript("FTOS_FlowSettings_Utills");
flowSettingCL.setFlowSettingVirtualCustomAttributes(formData.id,
"PropertyDetails")
    .then(function (r) {
```

in the "then" scenario of the async method will be added the entire script of *After Events*.

Client Script FTOS_FlowSettings_Utills

- contains one async function `setFlowSettingVirtualCustomAttributes` which will

- call endpoint FTOS_GetFlowSettings_ByLoanIdAndDigitalProcessorTypeName with two params: loanId and digitalProcessorTypeName for getting the flow settings
- use the result from getting the settings (previous point) and set them to virtual custom attributes on form (including digital journey id)

Server library FTOS_FlowSettings_Utils

functions:

- **getDigitalJourneyId** - returns digital journey id using loan id
- **getFlowSettingsParametersByLoanIdAndDigitalProcessorTypeId** - based on loanId and digitalProcessorTypeId and functions getDigitalJourneyId and getFlowSettingsParametersByDigitalJourneyIdAndDigitalProcessorTypeId, it returns digitalJourneyId, ProcessorSettingsId, settings and mapping from flow settings
- **getFlowSettingsParametersByLoanIdAndDigitalProcessorTypeName** based on loanId and digitalProcessorTypeName and functions getDigitalJourneyId and getFlowSettingsParametersByDigitalJourneyIdAndDigitalProcessorTypeName, it returns digitalJourneyId, ProcessorSettingsId, settings and mapping from flow settings
- **getFlowSettingsParametersByDigitalJourneyIdAndDigitalProcessorTypeId** - using digitalJourneyId and digitalProcessorTypeId, it returns digitalJourneyId, ProcessorSettingsId, settings and mapping from flow settings
- **getFlowSettingsParametersByDigitalJourneyIdAndDigitalProcessorTypeName** - using digitalJourneyId and digitalProcessorTypeName, it returns digitalJourneyId, ProcessorSettingsId, settings and mapping from flow settings
- **getFlowSettingName**: return flow settings name based on digital journey id

Server endpoint FTOS_GetFlowSettings_ByLoanIdAndDigitalProcessorTypeName calls function getFlowSettingsParametersByLoanIdAndDigitalProcessorTypeName from server library