HCL Universal Orchestrator 2.1
Portal Order processing Demo Pack

Scenario 1

The Business User's perspective

Workload Automation Technical advisor's team

Table of Contents

1. Scene 1: Cover

Welcome to HCL Universal Orchestrator.

Your mission, as a Business User, is to De-risk business and extend the outreach of automation. Performing this demo, you will learn how to deploy your HCL Universal Orchestrator cloud native orchestrator in minutes, orchestrate end-to-end business-critical processes, inter-connecting your flows with JSONata and restAPIs.

Steps:

1. Once your solution is deployed, access to the solution console using the credentials from Active Sandboxes pages.

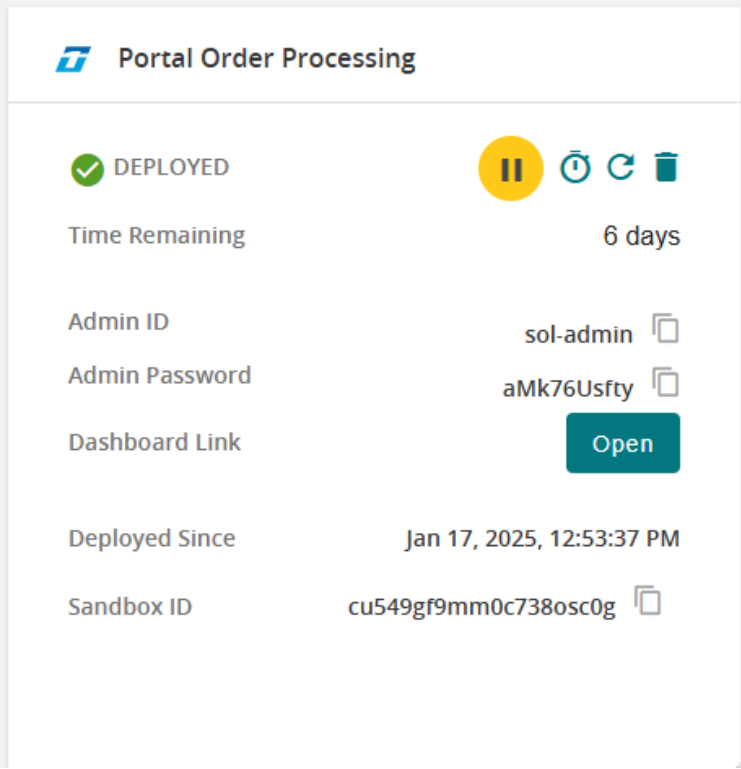

Figure 1 SoFy access

On Sofy' s console you can find the link and the credentials to access to the Dynamic Workload Console, our web interface, it's your single point of access to monitor your business-critical processes.

Steps:

1. Click on "Solution Content" menu and them "General Information" of HCL Universal Orchestrator tile.
2. Copy the User ID and Password
3. Open the Dynamic Workload Console link.

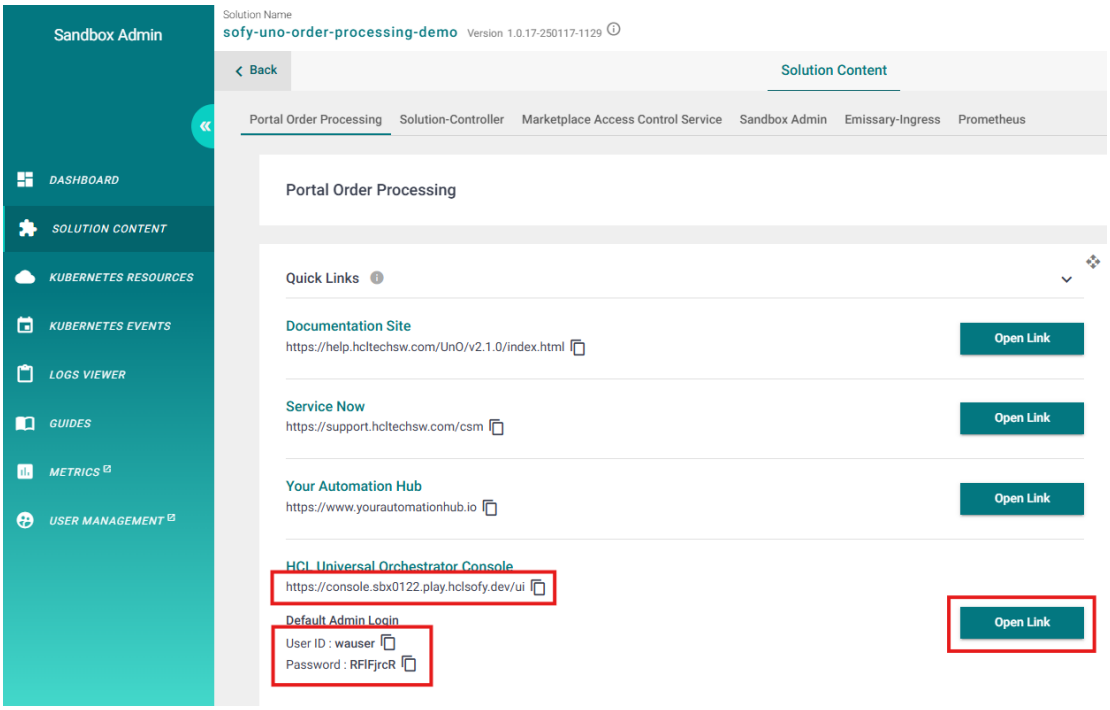

Figure 2 DWC credentials

Hint: If you may get an error message like this:

You should open the link via Incognito Window:

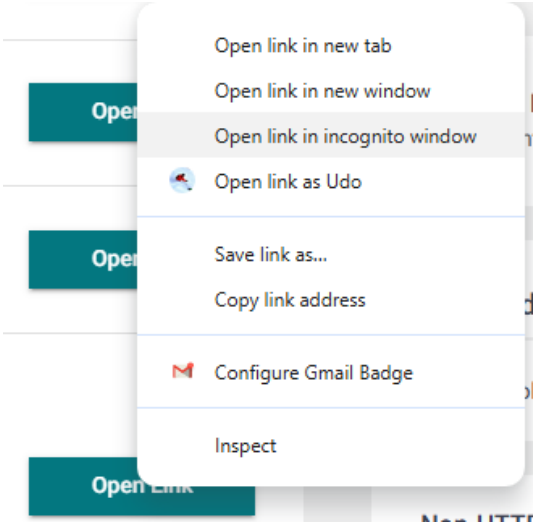

Figure 3 Open link in incognito window

2. Configure ocli (Orchestration command line interface)

During following steps we will import the “Retail portal purchase” demo scenario. While doing so we will also demonstrate our new command line “ocli”.

1- Under the user icon, click “Download Center”.

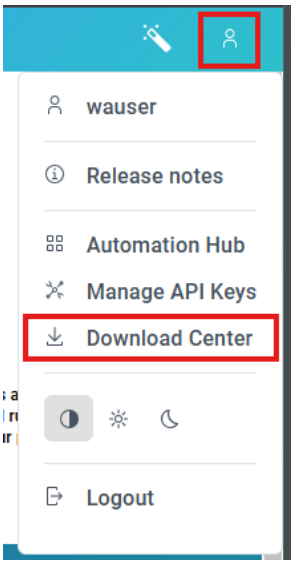

Figure 4 Download Center

2- Select the package for your operating system, in our case, Windows x86-64.

a- Once you click on the Windows tile, it will download the ocli OCLI_WIN_X86_64.zip.

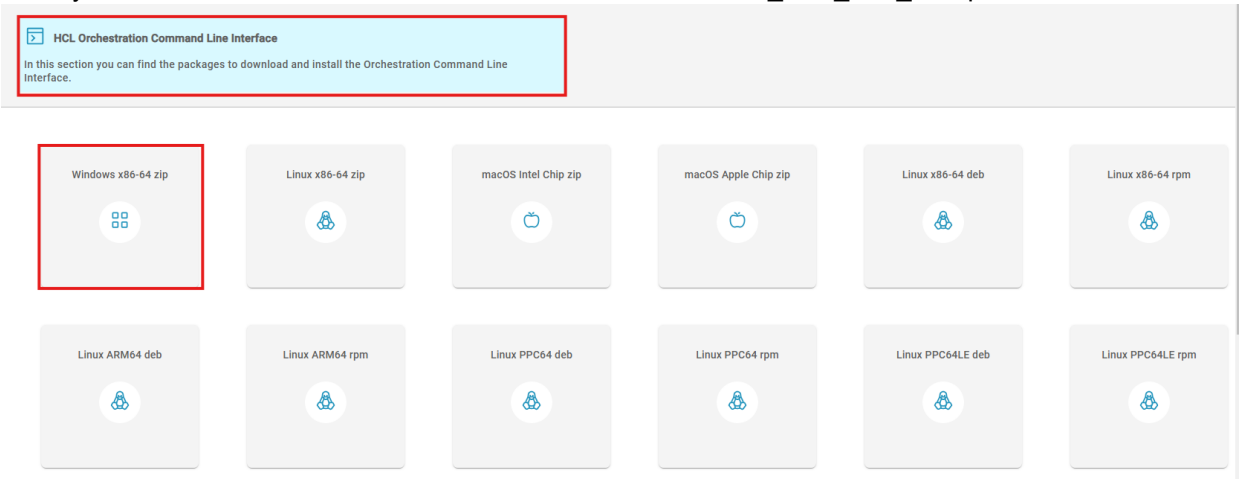

Figure 5 Download HCL Orchestration Command Line Interface

b- Save and extract the zip file, navigate to the extracted path, bin folder, and open the command line prompt. Hint: Press <Right Shift> and right click somewhere on the Explorer Window to directly open Power Shell for the correct path.

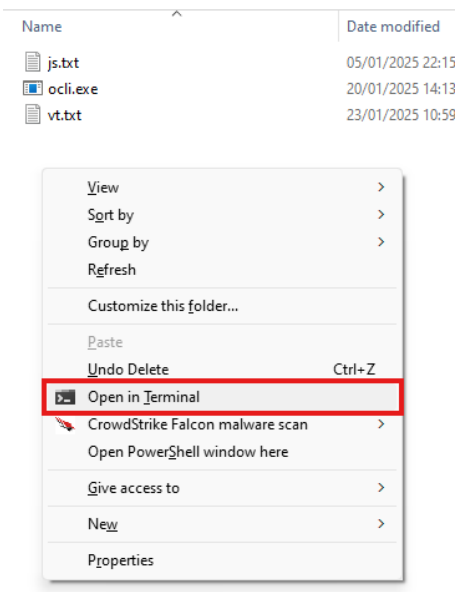

Figure 6 Open Terminal directly inside Windows Explorer

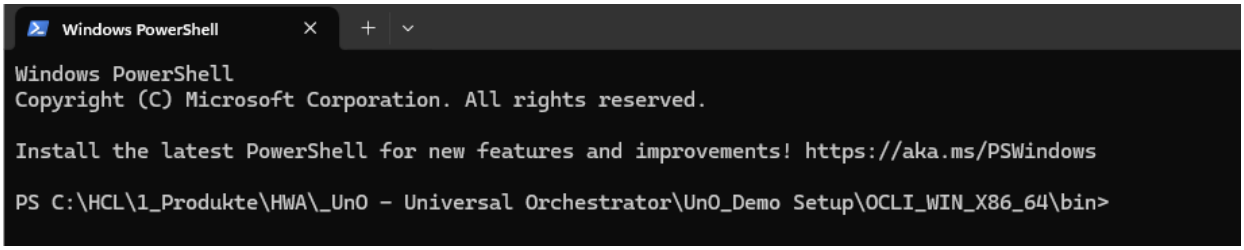

Figure 7 Power Shell Window opened

You may proceed now with Method 1 or Method 2 for creating YAML-file and API-key.

a. Method 1 to create YAML-file and API-key to upload demo scenario ("automatic")

1- Enter "*.\ocli.exe plan sc*" in Power Shell window

You will be asked for the gateway. Enter the gateway: e.g: "*console.sbs0168.hclsofy.dev*"
Hint: you find the gateway from your SoFy console:

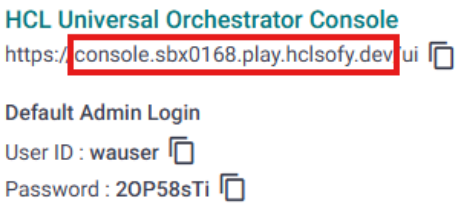

Figure 8 Selecting gateway from SoFy console

2- After pressing <Enter> the *port 443* is used as default (port 443 is correct for this scenario).
A link inside the power shell is shown to generate the API-key.

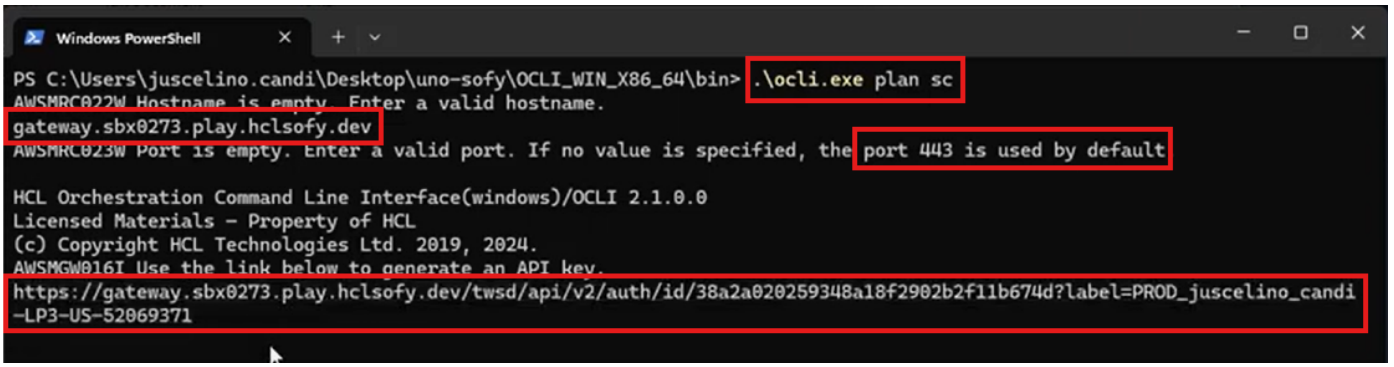

Figure 9 Creating Link to generate API key

3- Copy the provided link to a new browser tab and press <Enter>. The API-key is generated automatically.

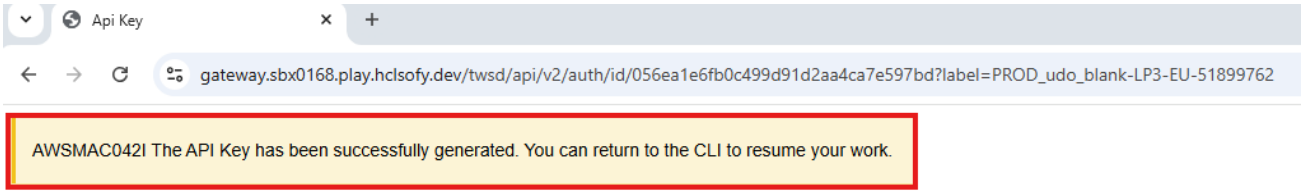

Figure 10 Enter link into browser tab to generate API-key

You find corresponding message inside the power shell:

You may proceed with chapter 3 now!

b. Method 2 to create YAML-file and API-key to upload demo scenario (“manual”)

First, let’ s create our api-key which will be used to connect with our UnO environment via OCLI.
On the user icon, select “Manage API Keys”.

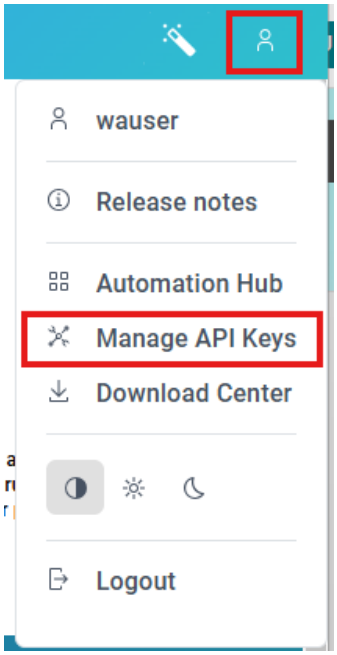

Figure 11 Manage API keys

1- Click “Add new”, Type Personal, and give a name, on our case “Ocli-access”

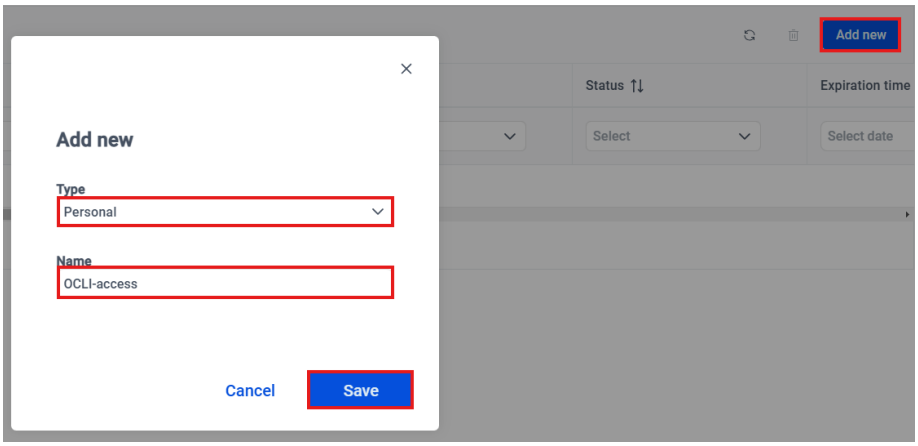

Figure 12 Create API-Key

2- Save the credential in a text file. (You won’t be able to access the key again).

3- Run “*.\ocli.exe plan*” to create the config.yaml file on the user’ s home directory.:
e.g. inside “*C:\Users\juscelino.candi\.OCLI*”

Figure 13 Create OCLI folder inside User directory

4- Open the config.yaml file and add the hostname of your SoFy instance (hint: without "*https://*" and "*/*" at the end), along with the port and the JWT token created on the step above.
Save the file.

Important: because of actual issue you may have also to edit the time zone: e.g. "*America/New_York*" (other time zone also possible: e.g.: Europe/Berlin).
Otherwise you may get error messages while executing commands later on!

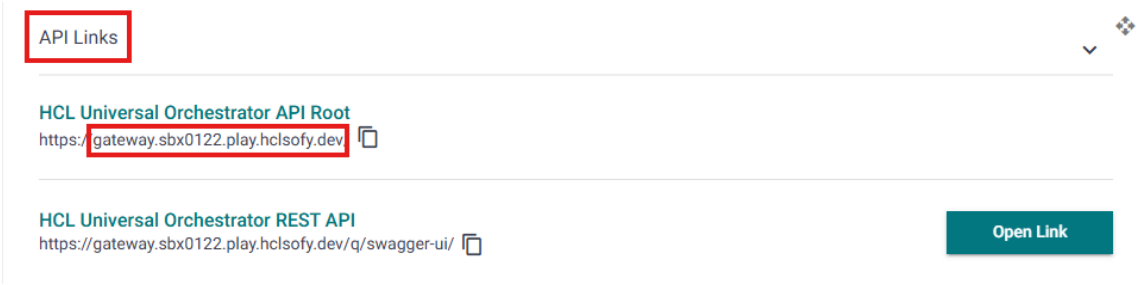

Figure 14 Gateway information on SoFy console:

```
connection:
# The name of the host that you want to connect to.
host: gateway.sbx0122.play.hclsofy.dev
# The tcp/ip port number that OCLI uses to connect to the specified host.
port: 443
# json web token
jwt: …eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9…
# If not specified, the time zone used is the one detected on the operating system.
local_timezone: America/New_York
…
```

Figure 15 Editing the YAML-file

Hint: you should download and install Notepad++ for easier editing

5- Validate the access by running ".\ocli.exe plan sc" to see the agents.

```
PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin> .\ocli.exe plan sc
HCL Orchestration Command Line Interface(windows)/OCLI 1.1.0.0
Licensed Materials - Property of HCL
(c) Copyright HCL Technologies Ltd. 2019, 2023.

CPUID              RUN    NODE         LIMIT FENCE DATE     TIME  STATE       METHOD
CLOUD               NA     UNIX CLOUD    100   100 08/21/23 15:39  L IOJ

PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin>
```

Figure 16 ocli plan sc

3. Upload demo scenario objects

Download from SoFy web page the UnODemo.zip which contains the Tasks/Workflow as well as Variable table we will use on the scenario, extract it at the same folder as the ocli.exe and run the bellow ocli commands.

PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin> .\ocli.exe model mkfolder /RETAIL
PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin> .\ocli.exe model replace vt.txt
PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin> .\ocli.exe model replace js.txt

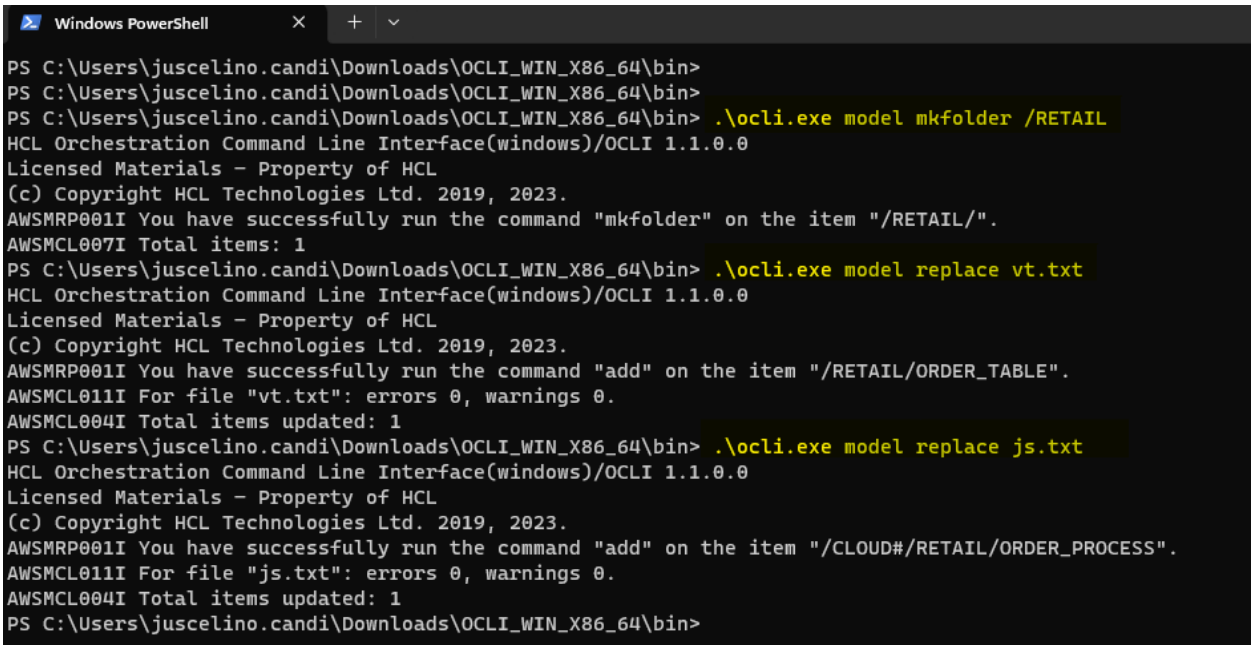

Figure 17 Commands to deploy UnO use case

To finalize we can submit the newly created Workflow.

PS C:\Users\juscelino.candi\Downloads\OCLI_WIN_X86_64\bin> .\ocli.exe plan sbs /CLOUD#/RETAIL/ORDER_PROCESS

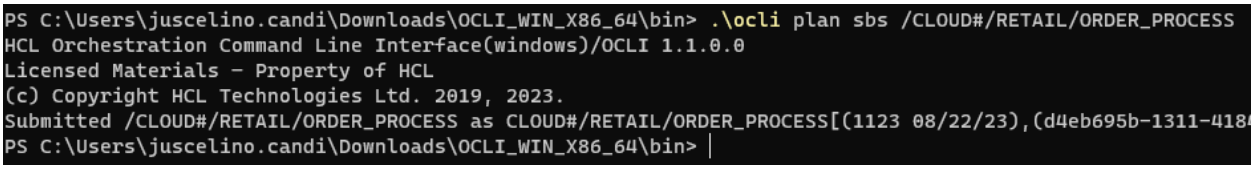

Figure 18 Submit use case

4. Visualize the demo scenario on UnO 2.1 UI

On Monitoring, select “Monitor”, we are able to see the Workflow submitted.

The new HCL Universal Orchestrator policy-driven adaptive plan makes your workload more autonomous, dynamic and scalable, which means there is no longer the concept of daily plan or the FINAL Workflow running every day.

The Tasks are added within the active windows (in which the dependencies of active Workflow instances are resolved and lunched). The plan extension and the old instances retention can be configured according your needs.

That means we will see the Workflows on the scheduled time frame, and they will be removed by UnO automatically without having to wait for the daily plan to run.

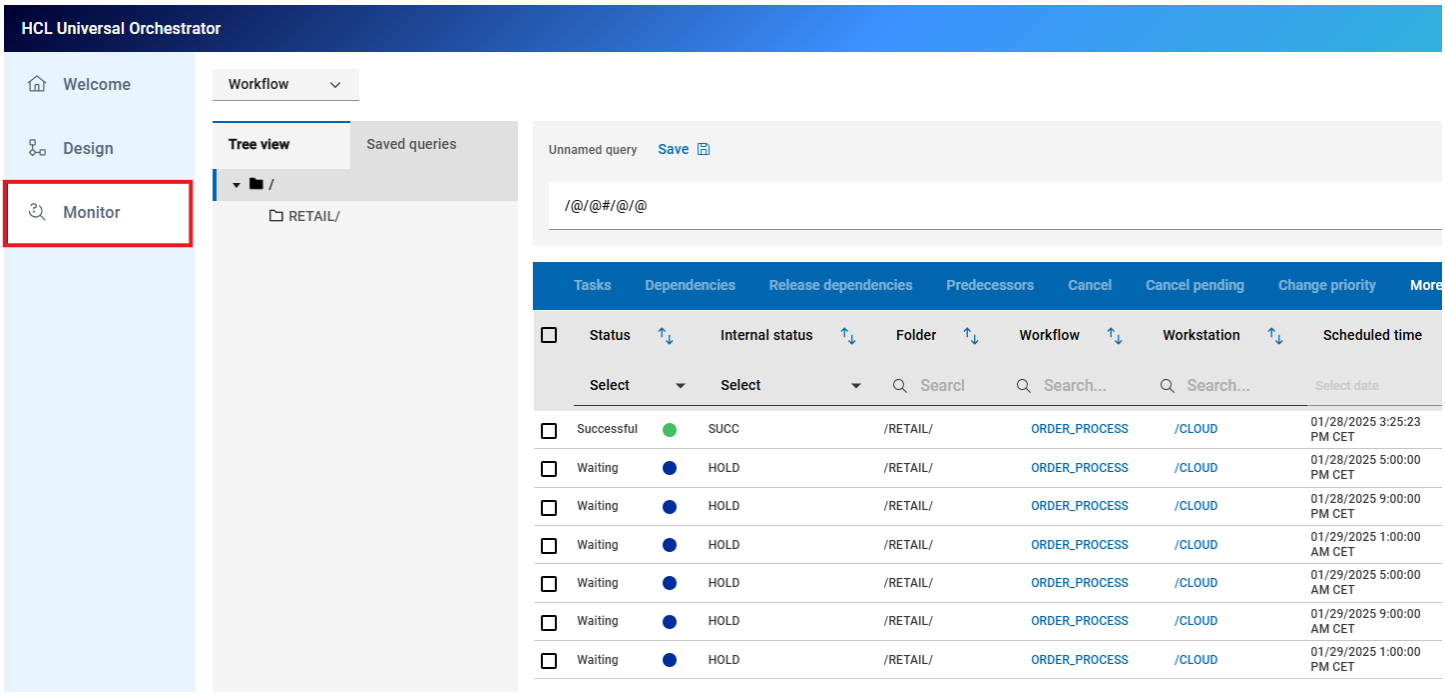

Figure 19 ORDER_PROCESS Workflow

After reviewing the Monitor view, you make the Workflow visible: Select “Design” → “Assets” → ”Workflow”:

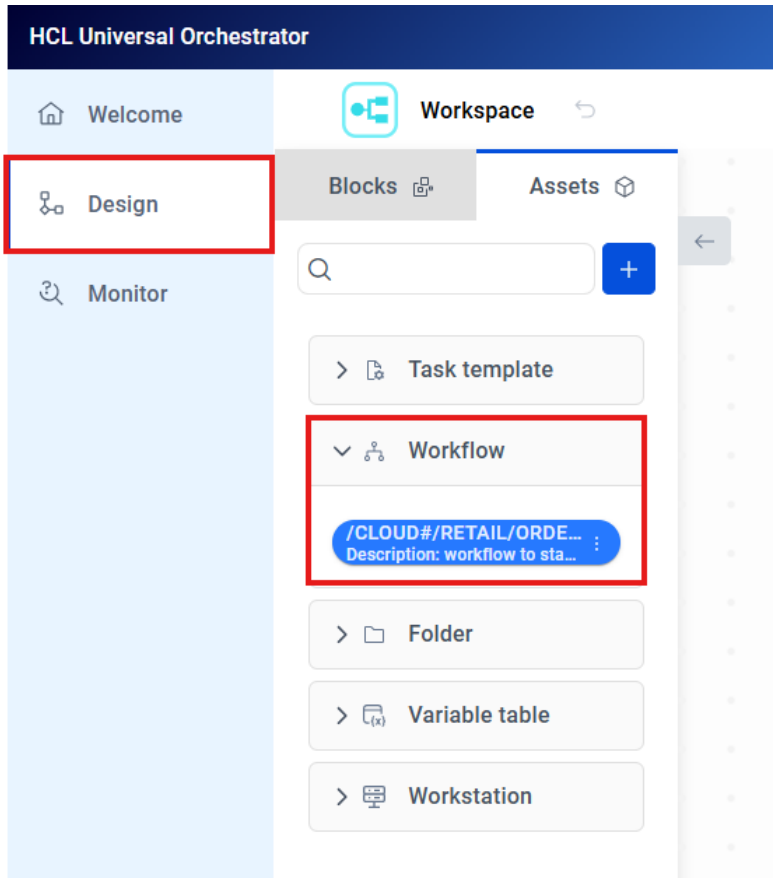

Figure 20 Select ORDER_PROCESS Workflow

Drag ‘n drop the Workflow to the canvas on the right:

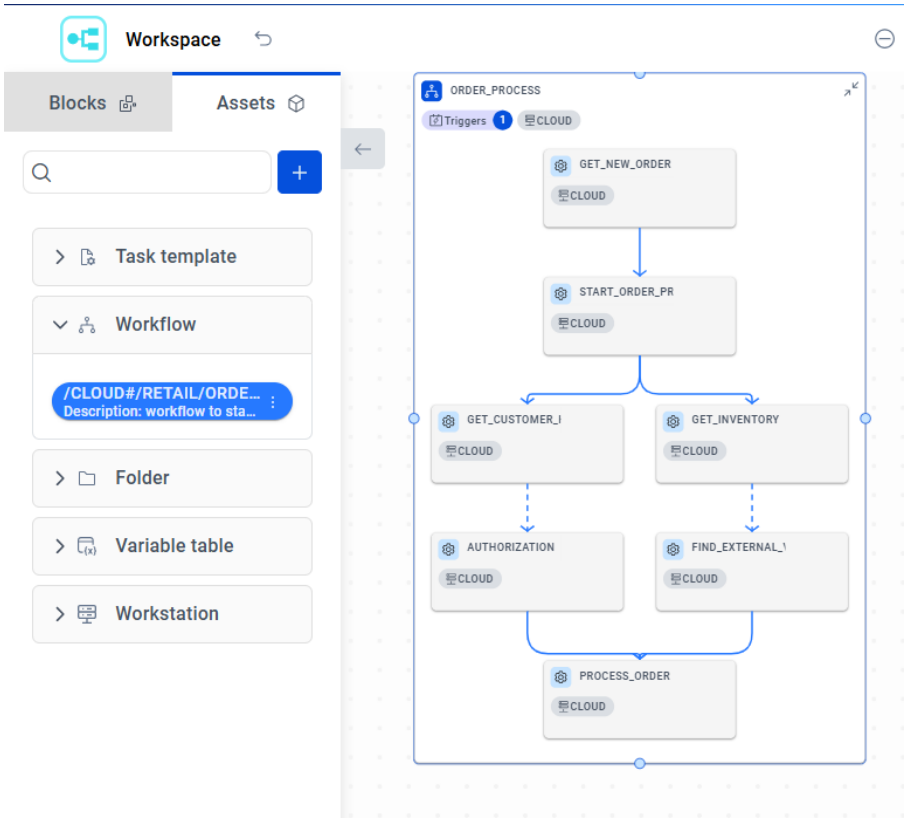

Figure 21 ORDER_PROCESS Workflow Workspace view

5. JSONata and variable passing

You can easily exchange data between Tasks and transform their output for using it in other Tasks. Task output data is combined into a single JSON file, containing the Task context. HCL Universal Orchestrator can query and transform information in the Task context using **JSONata4Java**, the Java version of **JSONata**, a lightweight query and transformation language for JSON data.

JSONata is used to extract relevant data that is hidden in potentially large JSON files using sophisticated queries with a compact and intuitive notation. It can be applied to virtually any problem that involves querying and transforming JSON data, and is able to perform the following actions:

- manipulate strings.
- combine and aggregate numeric data
- query and extract values
- create complex JSON output structures that enable complex data transformation tasks

For example, on the GET_INVENTORY Task we are using a JSONata function to get the order id, description and it’ s quantity. We need that information to make sure we have enough inventory and we are safe to process the order. All without using a single line of scripting.

For viewing the properties of a Task you have to select “Design” → “Assets” → “Task Template” and find the right Task (e.g. GET_INVENTORY). Inside the actual demo scenario it isn’t possible to select the properties directly from the Workflow, because they are NOT embedded.
Hint: use the search box to find it easier.
Click on the three dots on the right side of the Task and select “Edit”:

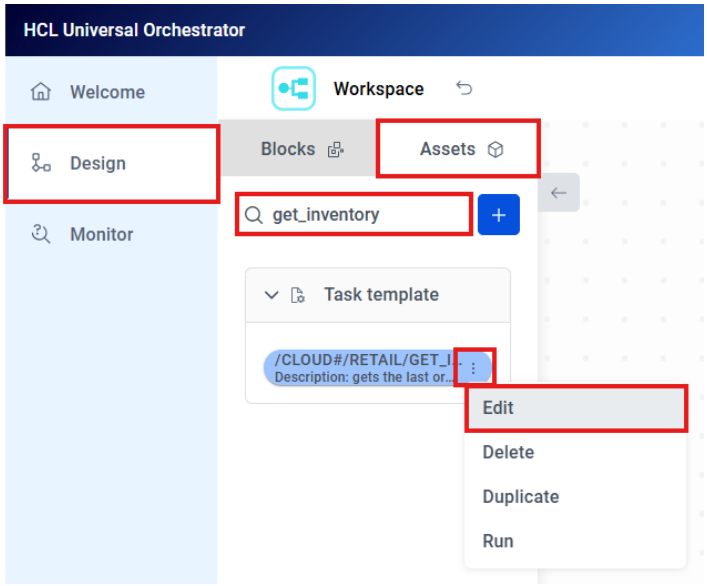

Figure 22 Edit GET_INVENTORY Task

On the right side of the screen the properties pane opens. While scrolling down you find next to the general information (Folder, Workstation, Type etc.) the detailed information for “Action” and “Body”.

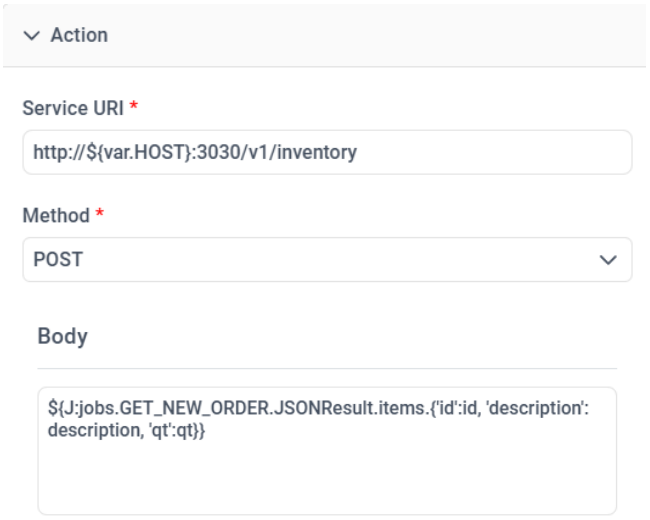

Figure 23 GET_INVENTORY Task properties

The last Task (PROCESS_ORDER) receives the JSONresult from the predecessors Tasks and process the order based on inventory.

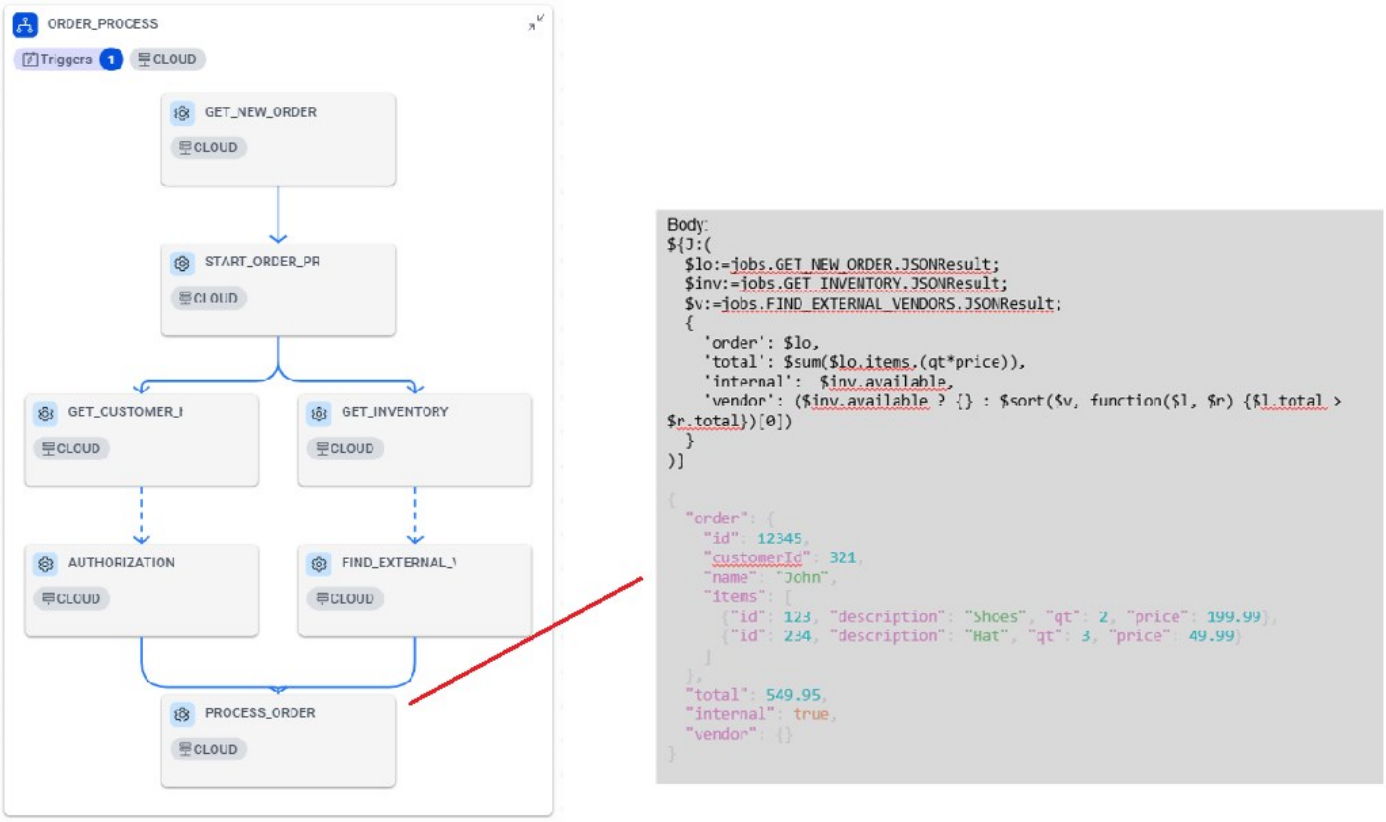

Figure 24 PROCESS_ORDER Task – body & result