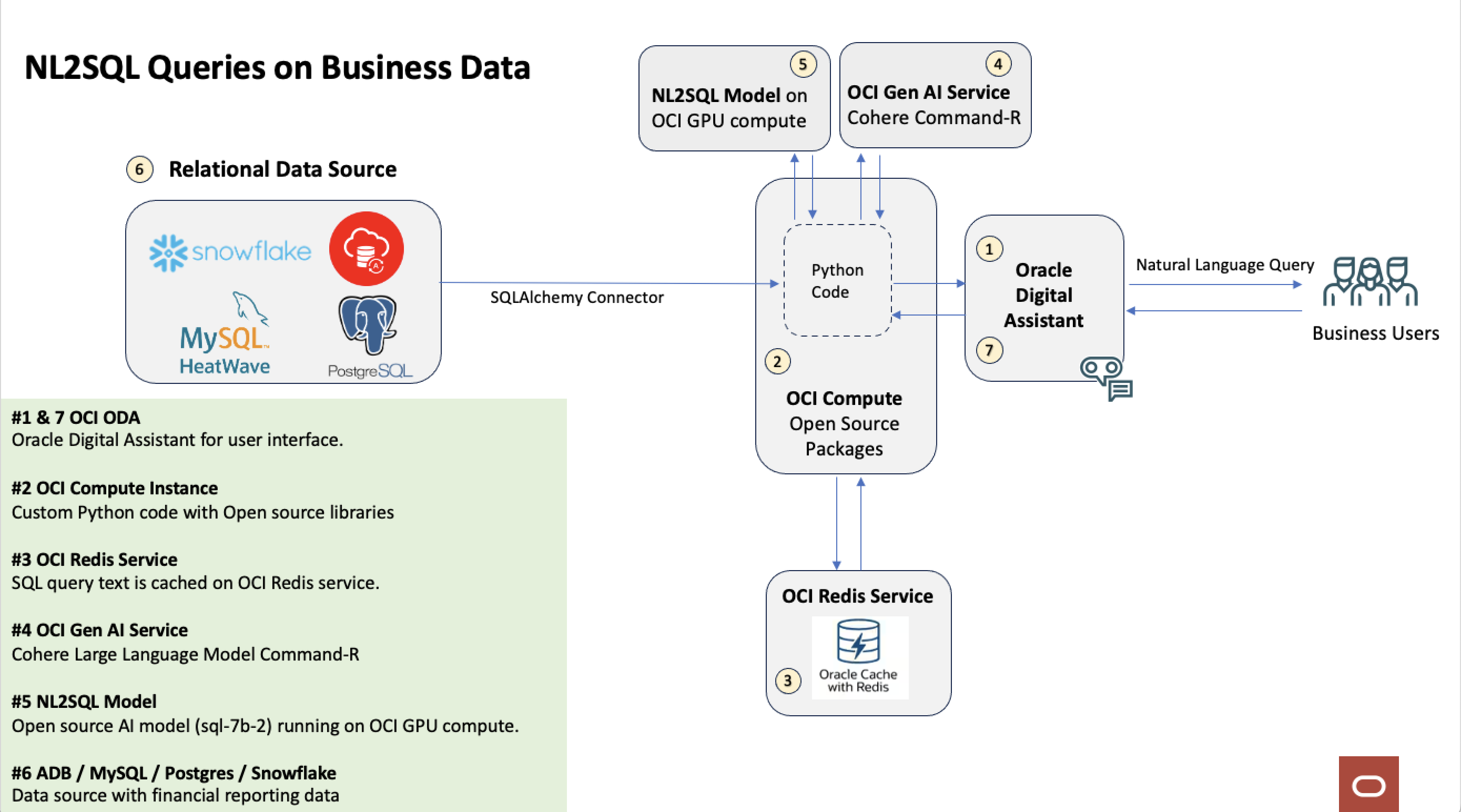
Install Instructions

This document provides detailed install instructions for a chatbot application that enables business users to make natural language queries on a variety of data sources (such as Oracle EBS, Peoplesoft etc.). Following visual depicts components that make up this solution:



OCI services that are provisioned as part of this solution are:

1. ADW (can be replace with a different relational database)
2. OCI Linux Compute Instance
3. OCI Redis Service
4. OCI Linux A10 GPU instance
5. ODA

Broadly the installation process will consist of:

* Components 1, 2, 3 and 5 are provisioned using Terraform.
* Component 4 will be provisioned via install steps provided in this document.
* A few environment specific configurations for different components will need to done manually.

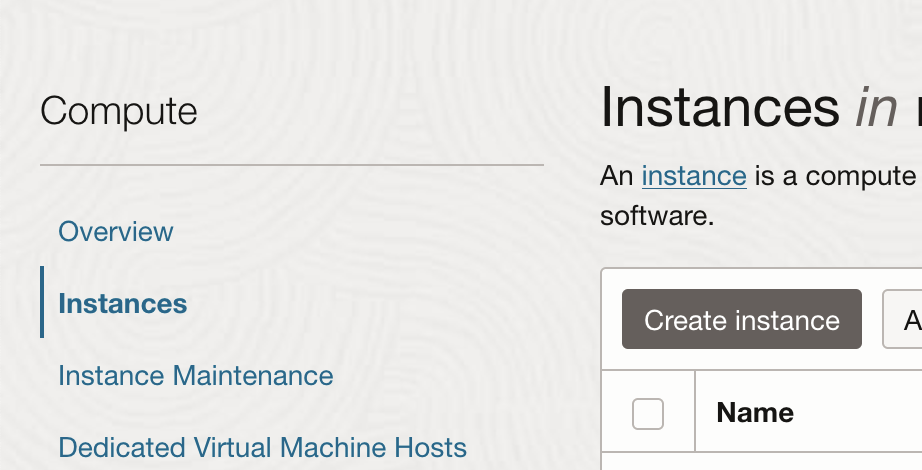
Step1: Use Terraform to provision ADW, OCI Linux Instance, Redis Cache and ODA.

Please refer to attached document.

Step2: Provision OCI GPU Compute using OCI console:

**Provision the Instance:**

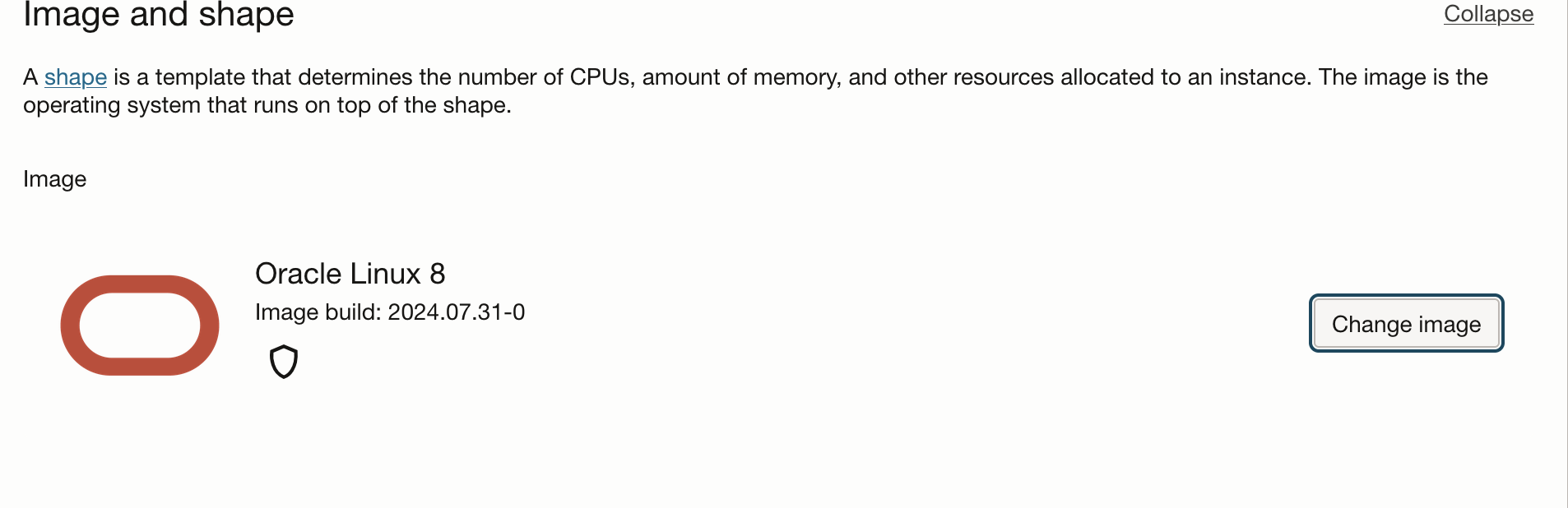
1. In OCI Console Navigate to Instances -> Create Instance:

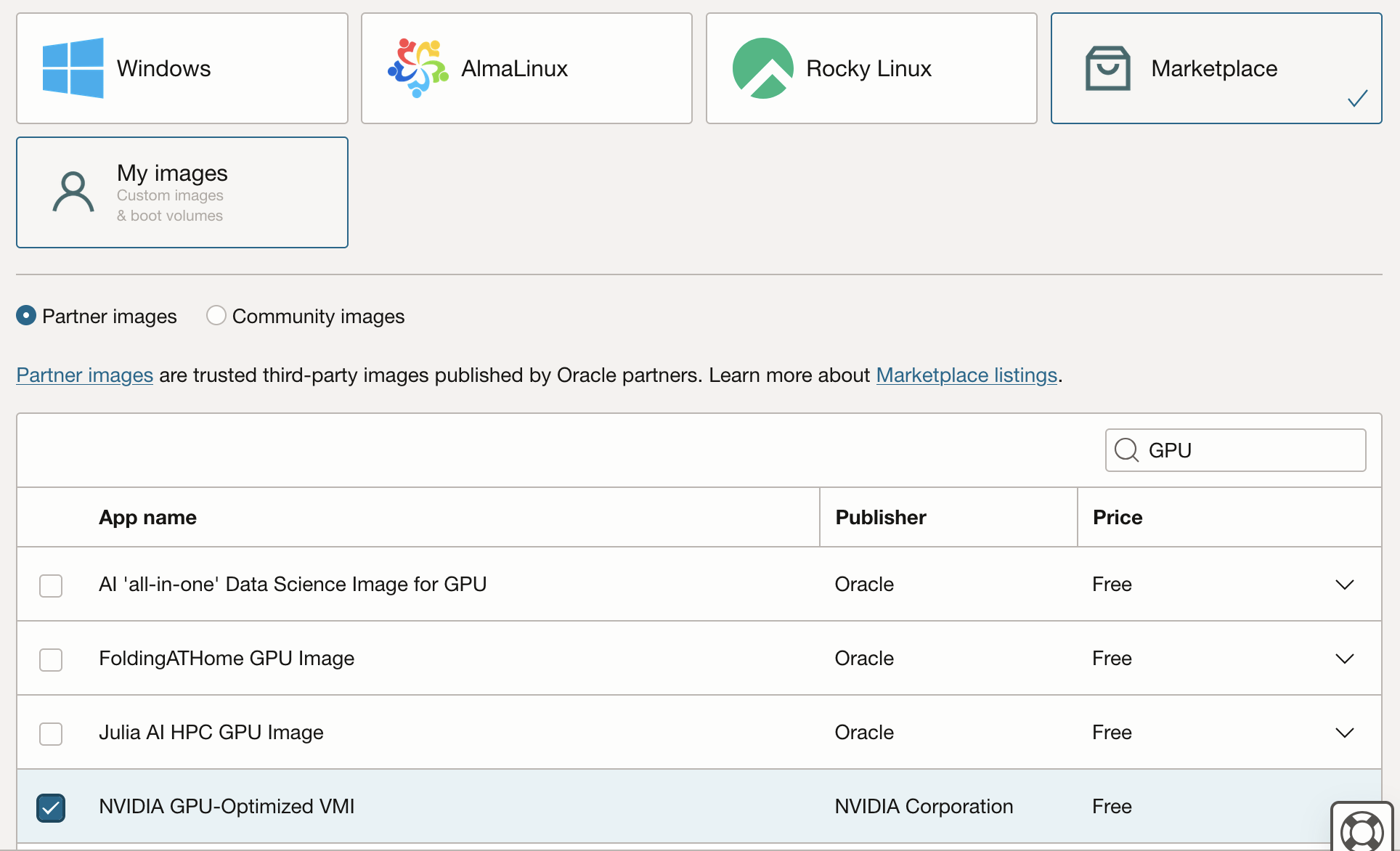


Fill out the create instance screen fully. You will need to provide VCN / Subnet details, upload your ssh keys and a few other details.

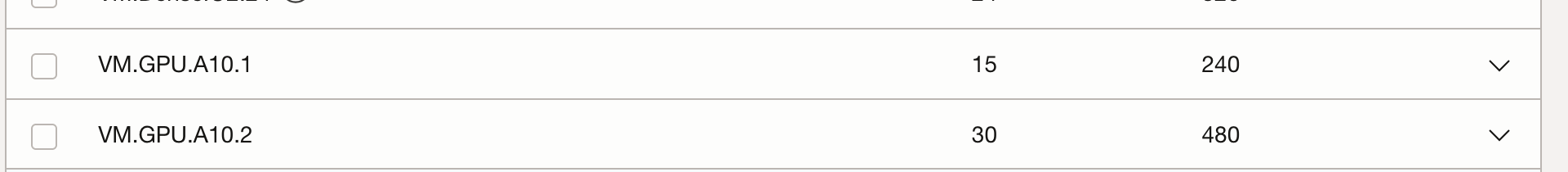
Pay attention to following sections to select the correct VM image and shape:

* Pick the Shape and Image - pick NVIDIA GPU VMI





* Specify the Instance Shape: pick A10.2 instance type:



* Specify any other parameters in the create instance screen and provision the instance.

1. Once the instance has been provisioned, ssh into the instance:

Using your ssh key, logon into the server.

****

Now install opensource libraries and custom code**:** Run following commands as ubuntu user (one command at a time)

sudo pip install torch

sudo pip install transformers

sudo pip install bitsandbytes

sudo pip install accelerate

sudo pip install sqlparse

sudo pip install -U sentence-transformers

sudo pip install spacy

sudo pip install uvicorn

sudo pip install fastapi

sudo python3 -m spacy download en\_core\_web\_sm

sudo apt-get install firewalld

sudo firewall-cmd --zone=public --permanent --add-port=8000/tcp

sudo firewall-cmd –reload

Copy following files from the application code into the instance:

Unzip the following file in the instance home directory:

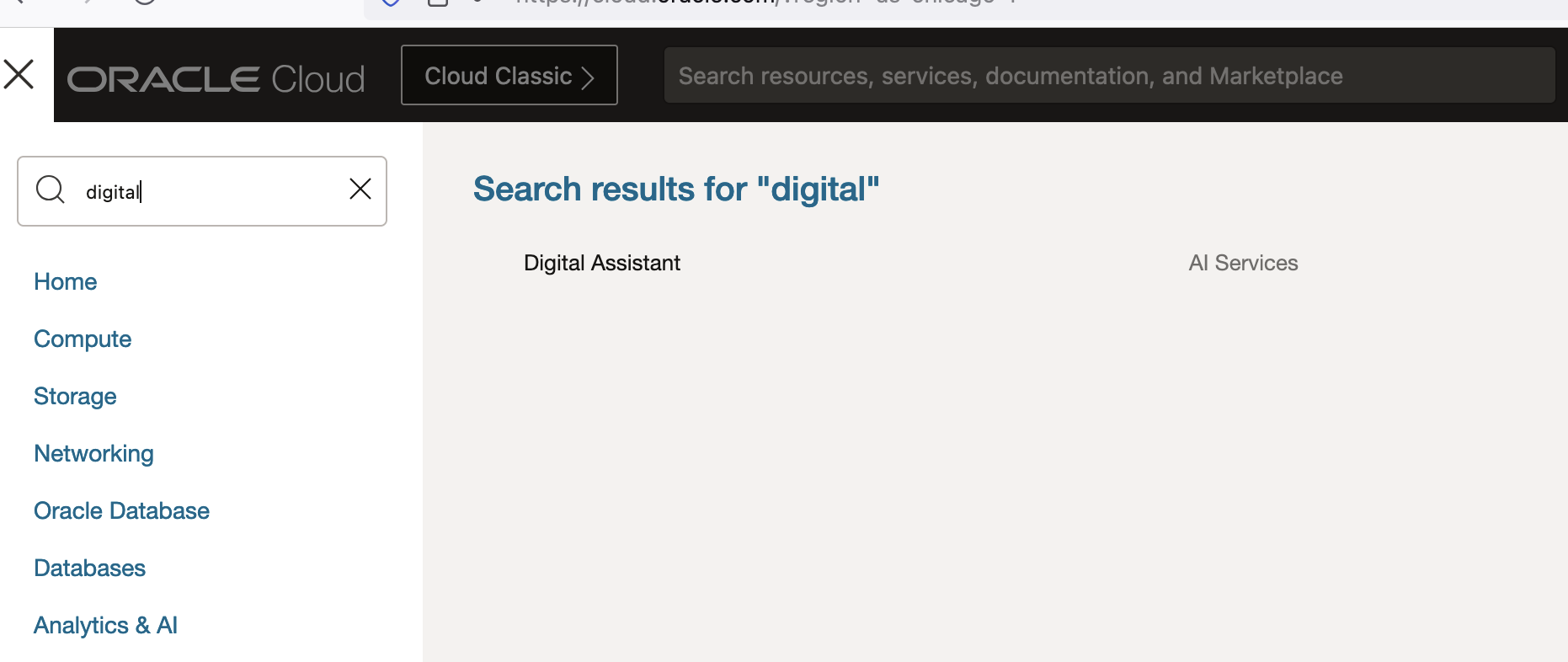
gpu\_py.zip

**Run the code:**

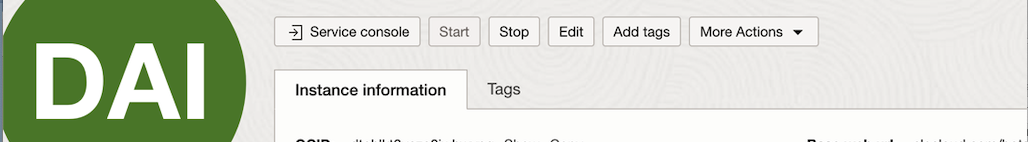
nohup sudo python3 api.py>log.txt &

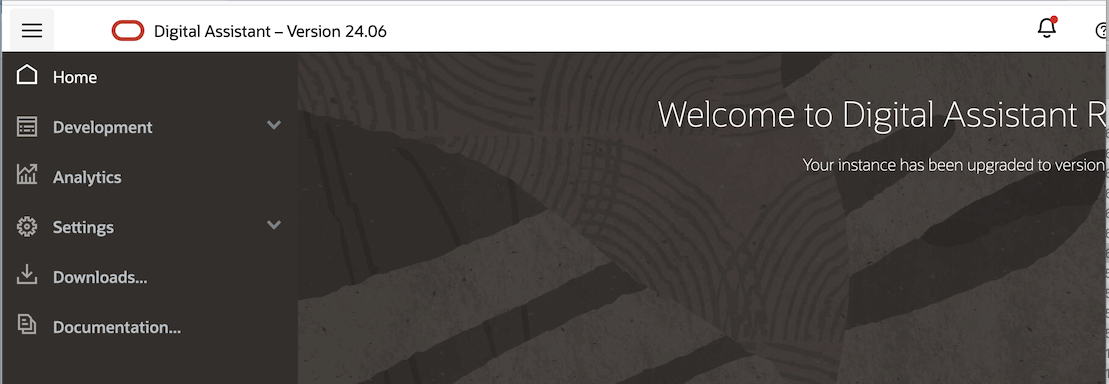
Step3: Perform Environment Specific Configurations:

* + - Configure ODA
      * Logon to OCI Console, search for Digital Assistant.

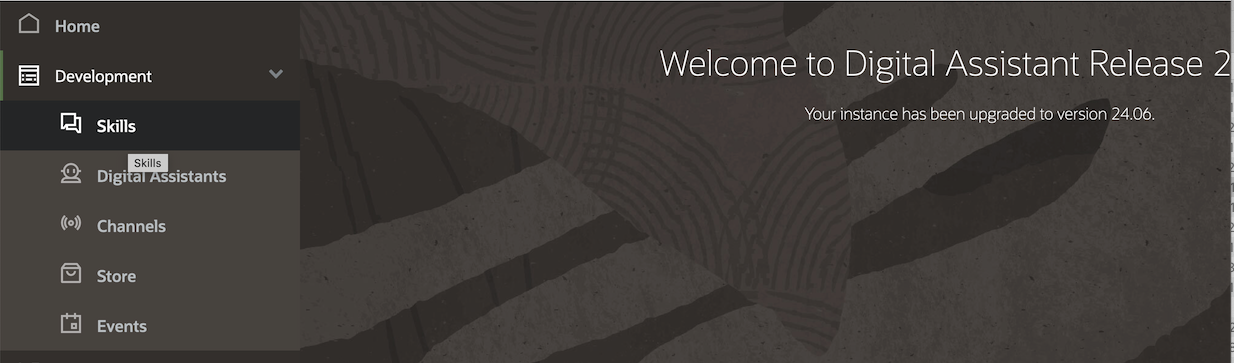


* + - * Locate the ODA instance that was created earlier using Terraform script and click on it.
      * Click on “service console”, it will open in a new browser tab:

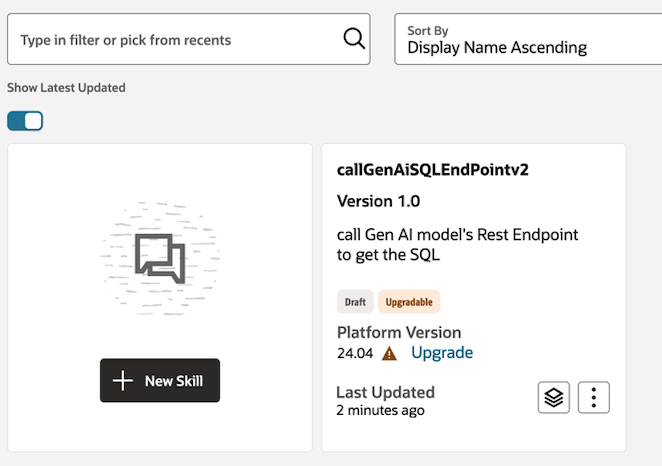




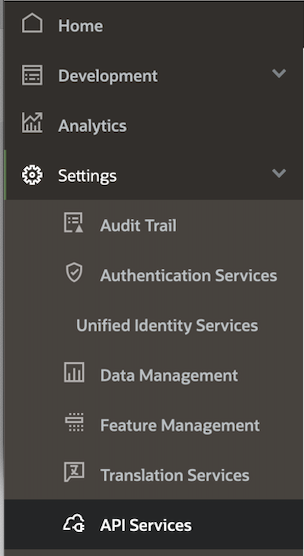
* + - * Click Development -> Skills



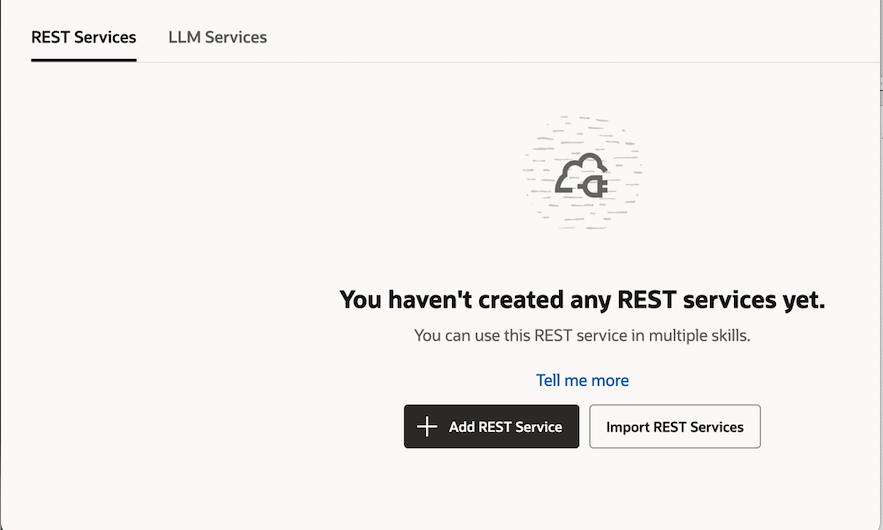
* + - * Click “Import Skill” (located right top corner)
      * Import the skill provided in the application code
      * After the skill is imported, you should see following under the list of skills:

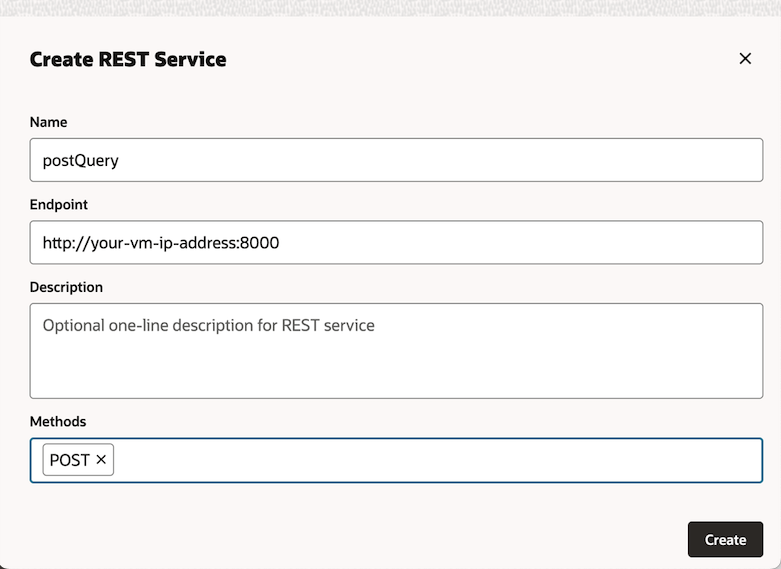


* + - * Next click on “settings”

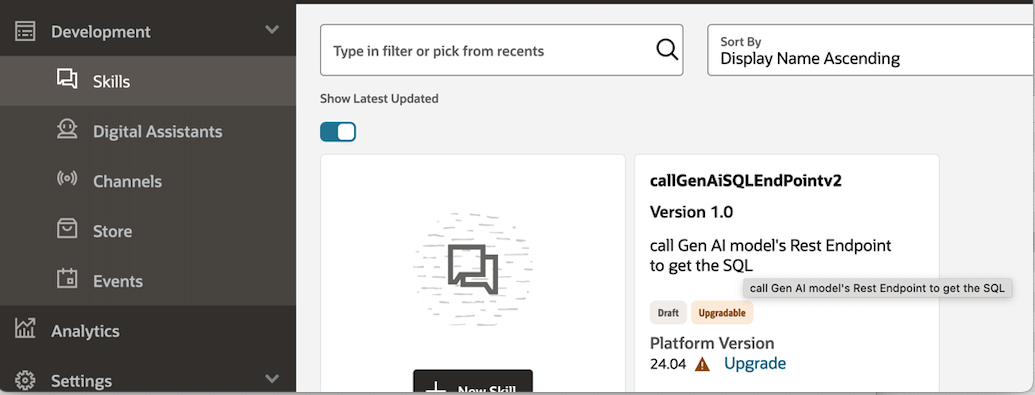


* + - * Click on API Services -> REST Services -> Add REST Service

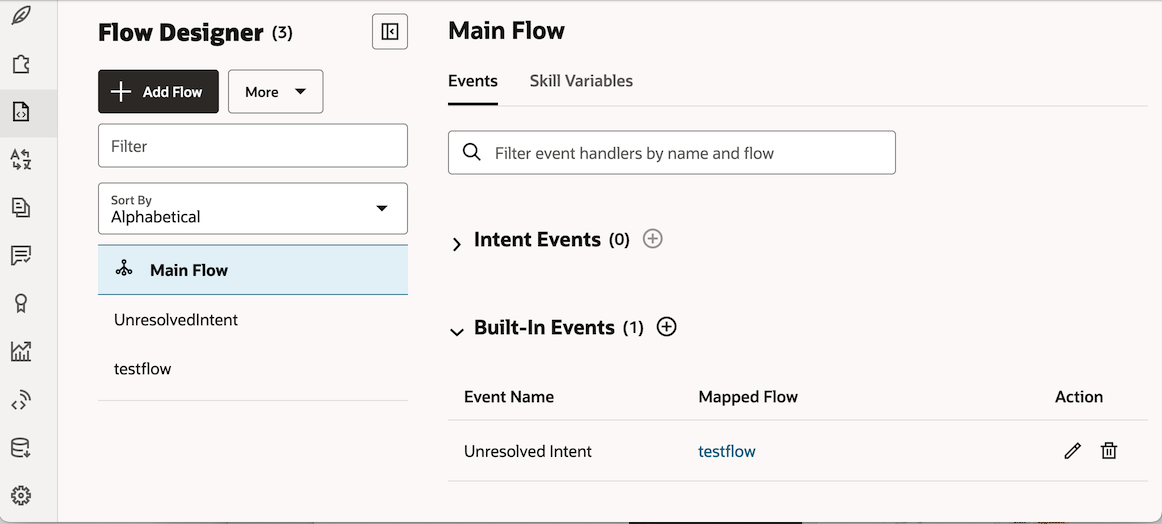




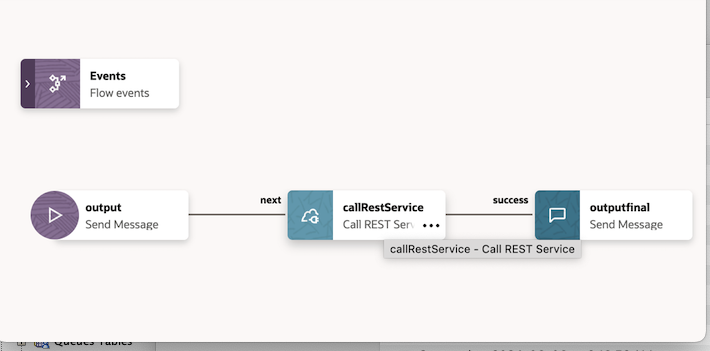
* + - * Next Navigate your Skill



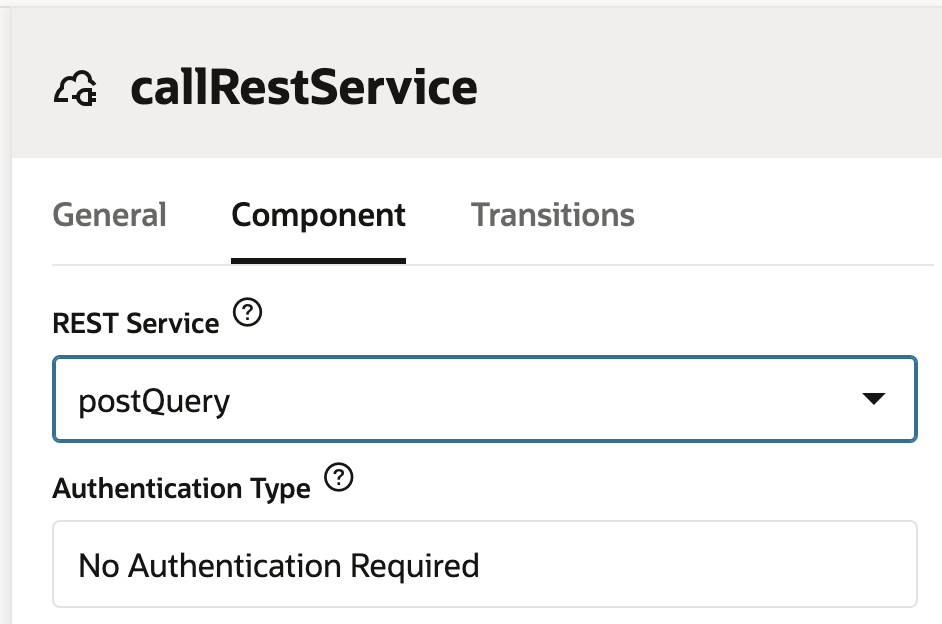
* + - * Click on the Skill to bring up the Skill details UI. Click on the third icon from the top on the left side

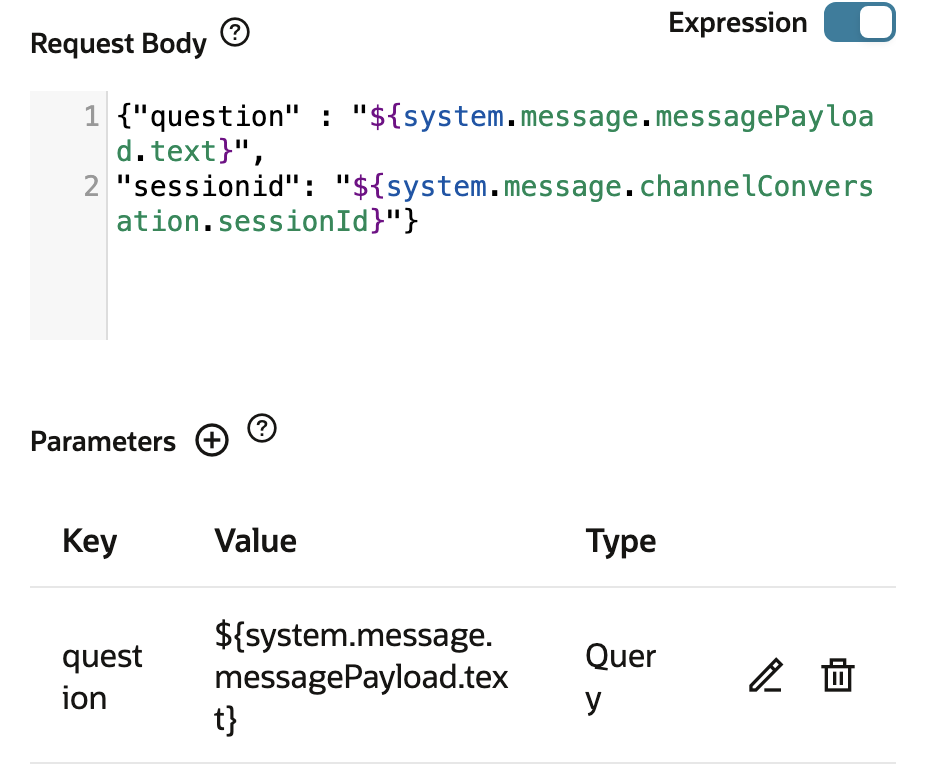


* + - * Next Click on “testflow”

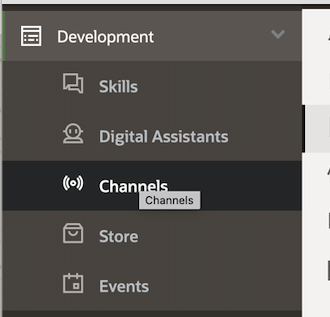


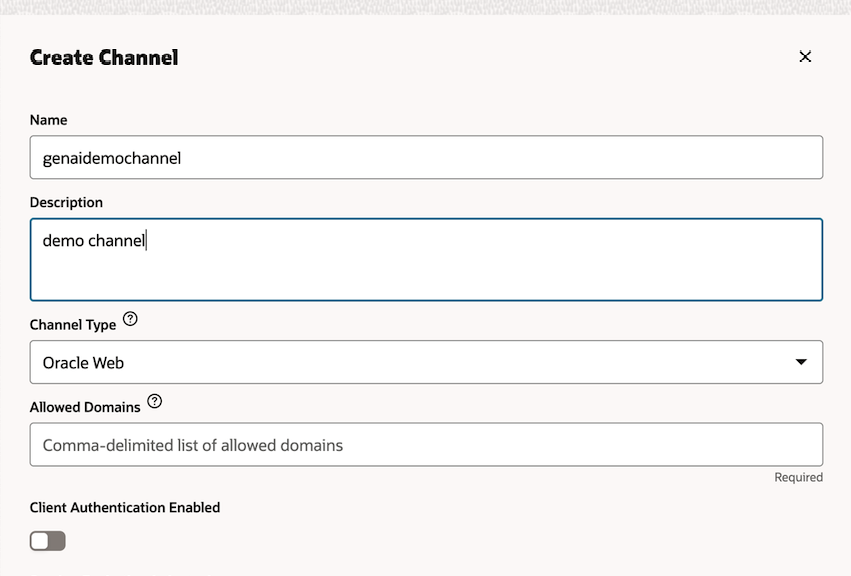
* + - * Next Click on callRestService: configure the REST call, pick the REST component created earlier.



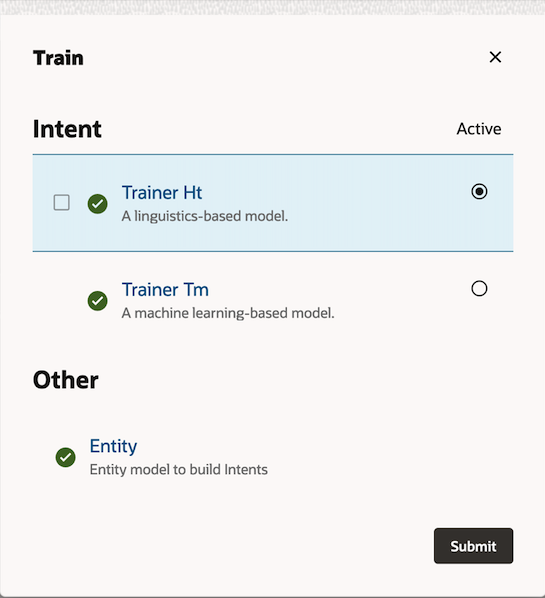


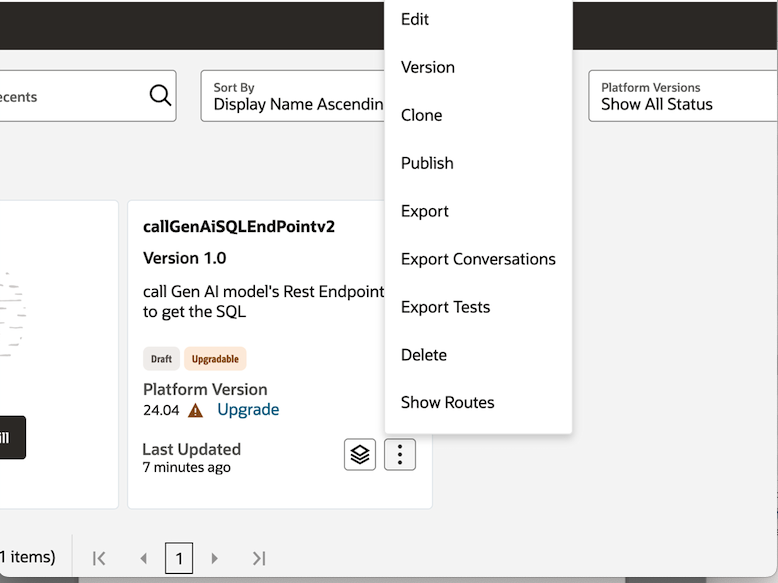
* + - * Create a Channel for your Skill



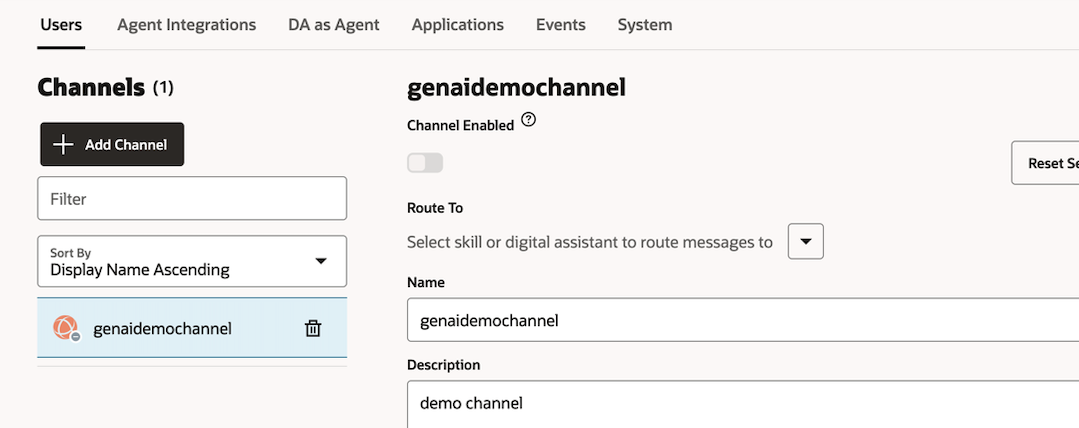


* + - * Train and Publish your Skill





* + - * Finally configure the Channel to use the published Skill:



* + - Configure environment specific application properties:
      * SSH into the Linux VM where the python application is running
      * Change Configuration.properties file located under /home/opc:

[DatabaseSection]

database.user=<your database user name>

database.password=<your db password>

database.dsn=<get from the wallet - medium/low>

database.config=<path where wallet is place on the linux server like /home/opc/wallet>

database.walletpsswd=Alm0stthere\*\*\*

[KeySection]

key.llm=notused

key.searchapi=notused

[RedisSection]

redis.url=rediss://<redis endpoint>:6379

url=<redis endpoint>

port=6379

[GenAISQLGenerator]

genaisqlgenerator.url=http://<nl2sql gpu host IP>:8000

[QueryResult]

max.resultset=25

filter.upn=N

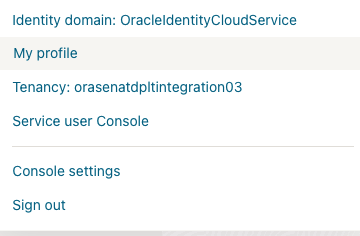
[OCI]

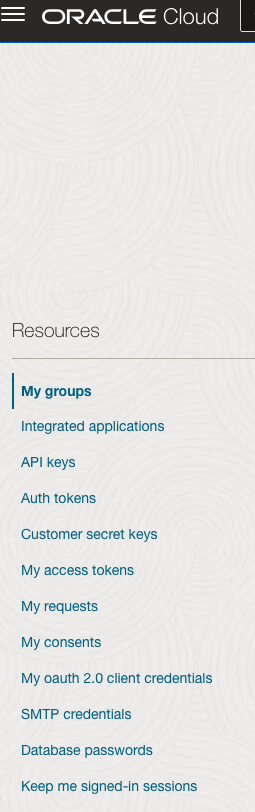
serviceendpoint.url=https://inference.generativeai.us-chicago-1.oci.oraclecloud.com

serviceendpoint.ocid=<your compartment ocid>

serviceendpoint.model=ocid1.generativeaimodel.oc1.us-chicago-1.amaaaaaask7dceyawk6mgunzodenakhkuwxanvt6wo3jcpf72ln52dymk4wq

* + - Add your OCI credentials
      * SSH into the Linux VM where the python application is running
      * Create a direct called “.oci” under /home/opc
      * Create OCI API signing key via OCI console – Click on the user icon located at the top right corner, then click on my profile, then API key (refer to the [link](https://docs.public.oneportal.content.oci.oraclecloud.com/en-us/iaas/Content/API/Concepts/apisigningkey.htm#two) here)





* + - * Make sure to save the public and private keys in the step above
      * Create a file called “config” under “.oci” directory on the linux instance
      * Update “config” file as follows:

user=<user ocid from API key>

fingerprint=<get from API key>

tenancy=<get from API Key>

region=<get from API key>

key\_file=<path to the pem file>

* + - Upload Wallet File



* + Download the wallet using OCI console from your ADW instance
  + Make a directory on your linux instance - /home/opc/wallet
  + Upload your wallet there and unzip it
  + Change sqlnet.ora file

WALLET\_LOCATION = (SOURCE = (METHOD = file) (METHOD\_DATA = (DIRECTORY="/home/opc/wallet")))

SSL\_SERVER\_DN\_MATCH=yes

* + - Setup Schema and test Data
      * Create the schema table

CREATE TABLE ACCOUNT\_PAYABLES\_TBL

(

VENDOR\_NAME VARCHAR2(4000 BYTE),

VENDOR\_NUMBER NUMBER(38,0),

VENDOR\_SITE\_DETAILS VARCHAR2(4000 BYTE),

INVOICE\_NUMBER VARCHAR2(4000 BYTE),

INVOICE\_DATE DATE,

GL\_DATE DATE,

INVOICE\_TYPE VARCHAR2(4000 BYTE),

DUE\_DATE DATE,

PAST\_DUE\_DAYS NUMBER(38,0),

AMOUNT\_DUE NUMBER(38,2)

);

* + - * Add data – use attached demodata.sql file
    - Run the application

nohup python v21\_nl2sql\_plt.py >log.txt &

* + - Access using ODA application

Unzip following file on your laptop:

websdk2104.zip

Open settings.js under samples directory

Provide following information from ODA console:

function getUri() {

return "oda-URI"

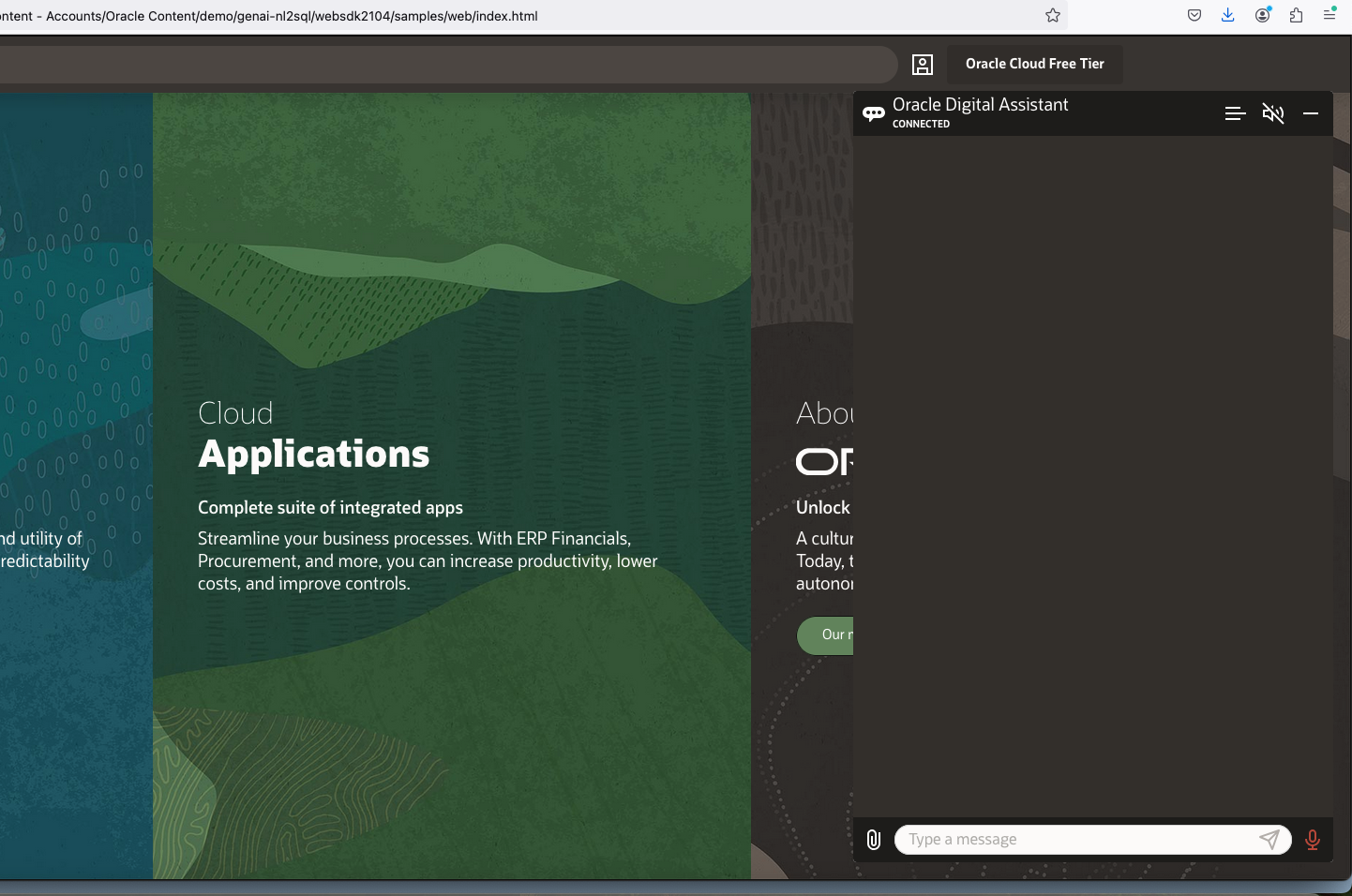
}

function getChannelId() {

return "channel id"

}

* + - * Finally, right click index.html, browse, refresh page the once:



Try following prompts:

Show me the invoices with highest amount due. Show invoice number and amount due and invoice date

Also show past due days

Show me total of all the payable invoices that are past due 0 to 30 days.

break it up by vendor name

For Advantage Corp show invoices with the highest payable amounts

show details of invoice# SE201039600

limit the output to vendor name, invoice#, amount due

create an invoice aging report with past due amounts grouped by vendor name, sum past due amounts 0 to 30 days, 30 to 60 days and 60 days or more

draw a bar graph for the previous data

Which vendors consistently appearing with overdue payables?

also show amount payable