ORACLE

Oracle Digital Assistant: Concierge Template Training

Creating bots by Business Users

maurits.dijkens@oracle.com

June 2023, Version 23.04

Table of contents

Before you Begin

This is hands-on lab in an entry-level exercise to start working with Oracle Digital Assistant using the Concierge template

Background

Oracle Digital Assistant is an environment for building *digital assistants*, which are user interfaces driven by artificial intelligence (AI) that help users accomplish a variety of tasks in natural language conversations. Digital assistants consist of one or more *skills*, which are individual chatbots that are focused on specific types of tasks.

In this lab, you will create a Q&A bot based on the concierge template. This bot will be able to:

- Welcome the user whenever the bot is started
- Answer specific questions directly
- Show different options when the question has not been clearly specified by the user
- Collect feedback from the user to understand if the bot is working fine

What do you need?

1. Access to an Oracle Digital Assistant instance
2. This document (**ConciergeTemplate.pdf**)
3. The concierge template (**Concierge_Template(23.04).zip**)
4. The Web SDK template (**web.zip**)
5. A QnA pdf (see below)

All the required assets are part of the zip file downloaded from GitHub.

Disclaimer

This is a description of an ODA template. The instruction and the example code are just examples and should be seen as a starting point of a project. For the latest documentation on Oracle Digital Assistant always check oracle.com.

Creating your Content

We are creating a Questions and Answers bot. For this, we need the questions with all the answers that we are planning to add. In a later section, we will see the different options to add this information to the bot, but for now, we suggest you to create a document with the following format:

Title for question? *Answer for the question* *Title for next question?* *Answer for next question*

When importing this document through the Knowledge feature, we need to consider that:

- Everything before a question mark (?) will be processed as a question. The answer will be everything after that, until a new line with a question mark (?) is found.
- The document must be a PDF file or a webpage with a public url.

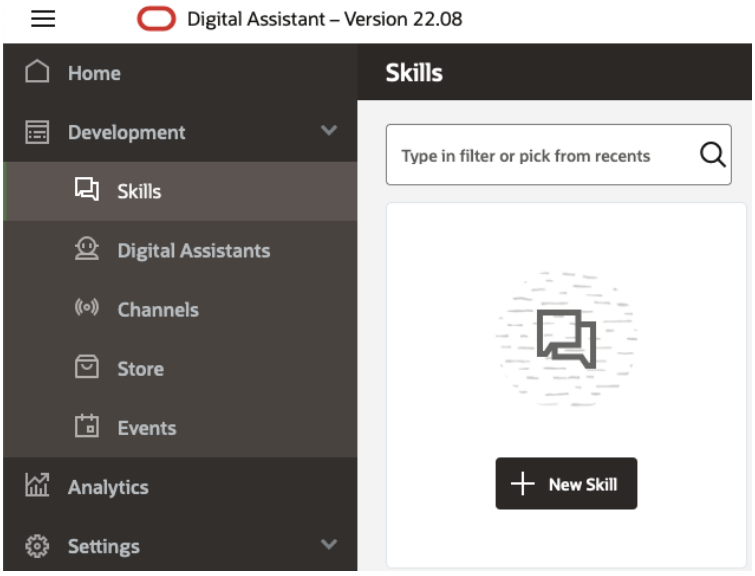
If your answer needs to include a question mark, you can always add it manually once the document is imported by editing your answer or you can create the intent manually. We will see both options later in this tutorial.

A **Sample_QnA.pdf** is provided in the download

Import the Template

We need to import the template as a new skill in our instance in order to use it.

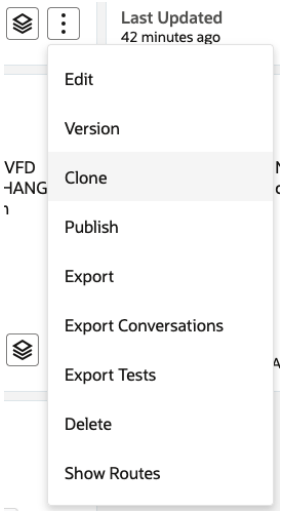
1. **Log in** into your Digital Assistant console
2. Click on the **hamburger menu** on the left hand side and click on **Skills**



3. Now **import** the template as a new skill by clicking in the Import Skill button on the top right corner

4. Select the **Concierge_Template(23.04).zip** file and click Open.
5. This may take some minutes to process.
6. Now you should be able to see the **Concierge Template** skill in your list of skills.
7. Let's **clone** the skill to modify a new version and leave the template as it is.

8. Click on the 3 buttons menu for the skill and select Clone



9.

10. Now let's add the information to this fields for the new skill:
    a. **Display name**: QnA Bot
    b. **Name**: QnABot (filled by default from Display Name)
    c. **One sentence description**: *provide a description that you feel suits best the skill*.
11. Your skill is now ready to use.

Adding Questions and Answers to the Bot

Let's add now the questions and answers that the bot will be able to manage. There are two ways of doing this: using the knowledge document feature (currently only working in English) or adding them manually.

Knowledge document

If you already have a web page or PDF document (**Sample_QnA.pdf** provided) with question and answer pairs, you can use the Knowledge feature to ingest those Q&A pairs from the document and generate answer intents automatically. (Any text in the document that does not follow the question/answer format is ignored.) When you create answer intents this way, example utterances are also generated for the intents.

To generate answer intents from a question and answer document:

1. Click the **Knowledge** icon in the left navbar.
2. Click **+ Knowledge Document**.
3. In the **New Knowledge Document** dialog:
    a. Specify a name (short single word like '`faq`') and language for the document.
    b. Select **PDF** and upload the document or select **URL** and provide the URL for an HTML web page. If the document is a PDF, select the checkbox acknowledging that it will be temporarily stored.
4. Click **Create**.
5. Wait for the generation of the answer intents to occur. This might take a few minutes.
6. Once the job is completed, click **Review Intents** to go over the generated intents and training utterances. Pay particular attention to each question and answer to make sure that each contains the right text.
7. To edit an intent's name, question, answer, or utterances, click its **Edit** icon.
8. For intents that you don't want added to the skill, clear the **Include** checkbox.
9. Click **Add Intents to Skill** to add the generated intents to the skill.

10. In the left navbar, click **Intents** and make any further adjustments to the intents, such as changing the conversation name and adding further example utterances.

Adding them manually

If you need just a few answer intents, you can create them manually.

1. Click **Intents** in the left navbar.
2. Click **Add Intent**.
3. Click to enter a descriptive name or phrase for the intent in the **Conversation Name** field.

IMPORTANT: The conversation name for each field should be filled with the actual question of the intent since it will be shown in the bot.

4. Add the intent name in the **Name** field. If you don't enter a conversation name, then the **Name** field value is used instead.

IMPORTANT: In order to make the answer intent work with the template, make sure the name includes ***.ans.*** (example: faq**.ans.**somename).

5. Add an answer to the **Answer** field.
6. In the **Examples** section, add training utterances that reflect typical ways that users would express the question that the intent is answering.

Every time we added or changed Intents, we need to (re)train the NLP-engine by clicking 

and selecting the training model Ht (quick; only keywords; for development) or Tm (slow: Real deep learning NLP for test/production).

Let's **test** now the Bot to get the answers! As we did before, click on the Preview button and start typing questions for the bot. You can start with a greet (like Hi or Hello), and then ask any of the questions we've created.

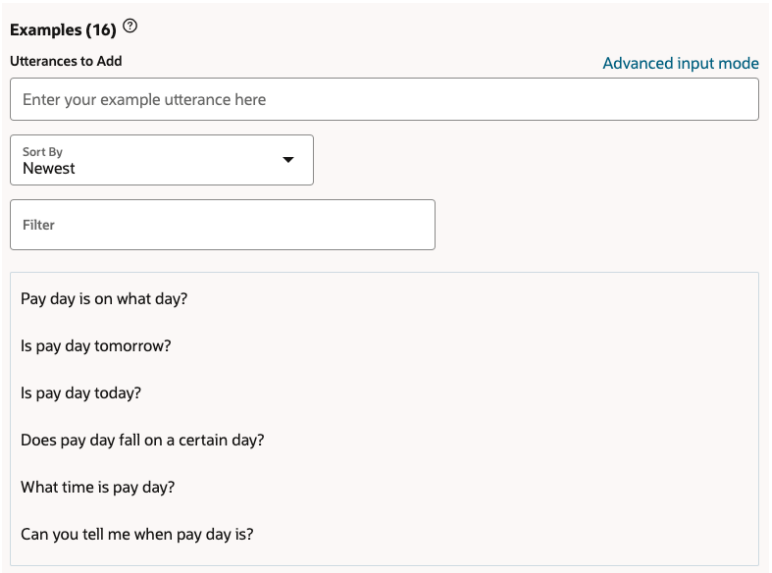
Additional Training

One of the keys to success of a bot is training it. We need to let the bot know different ways on how the user may ask for some information. Human language is complex. We can refer to the same thing in a lot of different ways.

We have different options to achieve this

Adding manual utterances

One is by adding manually whatever utterance we believe are useful. We can do this in the details of each of the intents, under the Examples sections.



Data Manufacturing Tool

As a single developer, it can be difficult, or even impossible for you to create a large, varied set of utterances. Rather than trying to come up with training data on your own, you can use Oracle Digital Assistant to crowd source this task. Assigning this to the crowd can be particularly useful when you need utterances that only experts in the application or the domain can provide.

Data manufacturing jobs are collections of tasks assigned to crowd workers. The jobs themselves focus on various ways of improving intents.

You can learn more about this here.

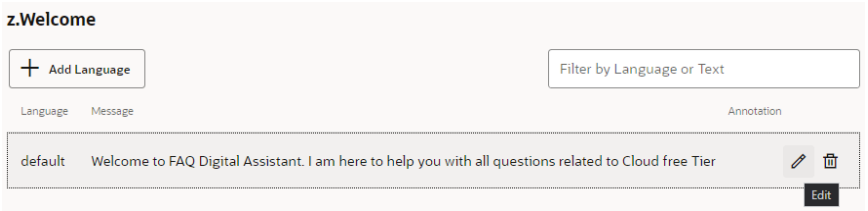
External tools

There are external tools that can help you create the utterances. One example would be WordTune.com

Welcome Message

The entry point for this skill happens when the user greets the skill. This has been done for you in the template and you only need to modify the message and the image that are being shown. This information is being stored in the *Resource Bundles* **z.Welcome** and **z.WelcomeImageURL**. In order to modify then:

1. Go to the **Resource Bundles section** to click in this icon in the left navbar.
2. In the list of keys, go to the **second page** and **select z.Welcome**.
3. Hover over the message you see with your mouse and click the pencil to **edit** it.

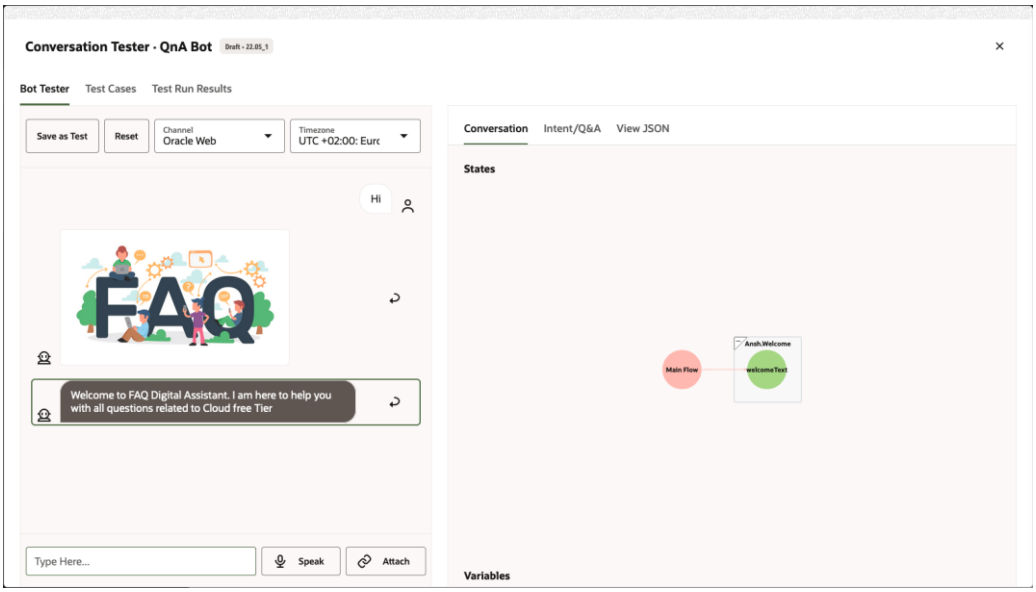


4. **Modify** the message field with the text you'd like to show as a welcome message.
5. Once you're done, click on **Update Entry**.
6. Now click on the **z.WelcomeImageURL** resource bundle and follow the same logic to modify the URL. By default, the image is looking like this:

7. You can leave it as it is or find a new one.

Now that we have modified this information, let's see how it looks by previewing the skill.

1. Click on the Preview button on the top right corner.
2. Type "Hi" in the chat.
3. You will see the answer in the Bot Tester section. For now, ignore the rest of tabs and sections in this screen. You will learn this later.



Unresolved Intent

Whenever a user input (utterance) does not match any of the questions we have in our bot, this will be handled by the unresolvedIntent. The easiest way to proceed is to show a message that informs the user that the input was not understood and needs to be rephrased. By default, the message looks like this:

Sorry, I didn't get that. Can you try again with different wording?

This message can be edited like this:

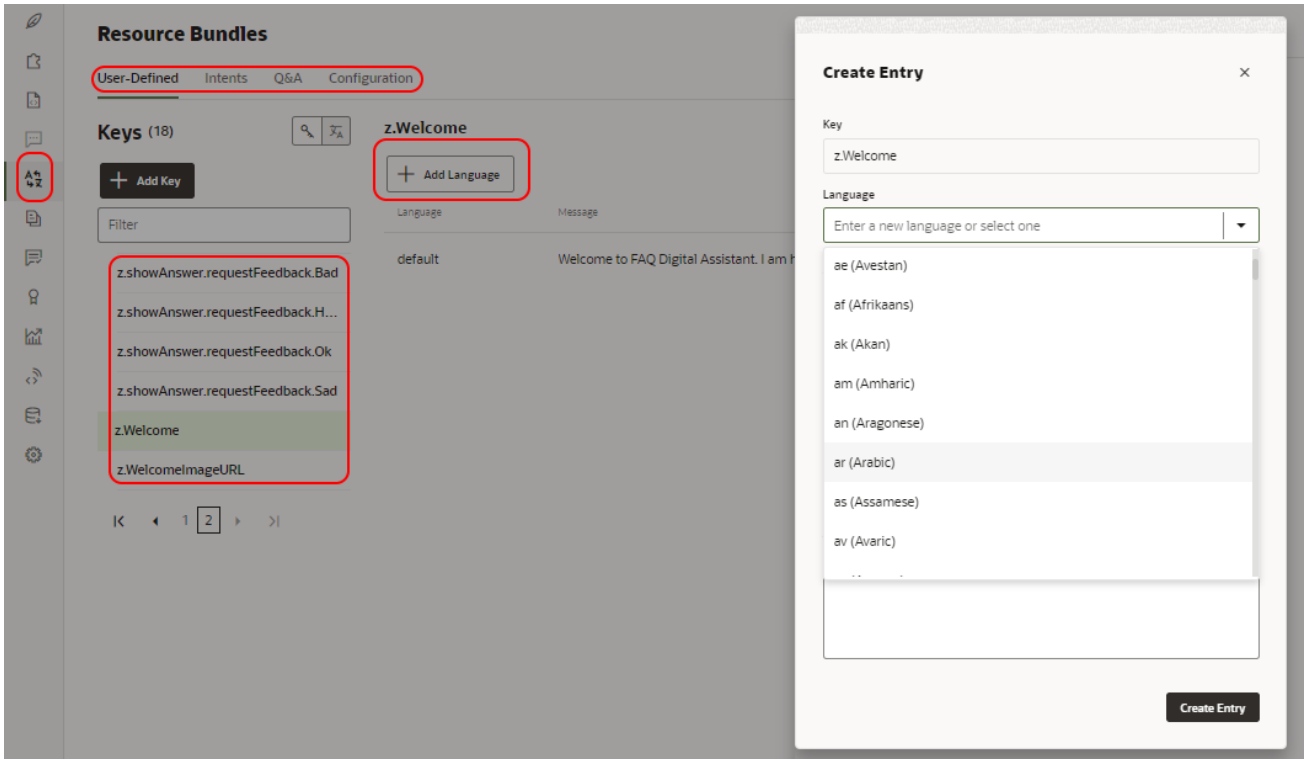
1. Go to the **Resource Bundles section** to click in this icon in the left navbar.
2. In the list of keys, go to the **second page** and select **UnresolvedIntent.unresolvedMessage**.
3. Hover over the message you see with your mouse and click the pencil to **edit** it.
4. **Modify** the message field with the text you'd like to show.
5. Once you're done, click on **Update Entry**.

Adding Languages

Adding languages to your skill is very easy:

1. Go to the **Settings section,** click on this icon in the left navbar.
2. Scroll down, click 'Add Language' button and add a language:

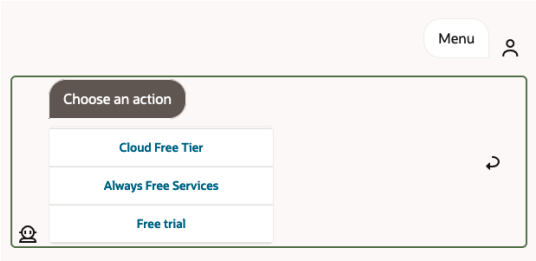
3. Go to the **Resource Bundles section,** click on this icon in the left navbar.
4. For each of the messages used, you can add the translation:



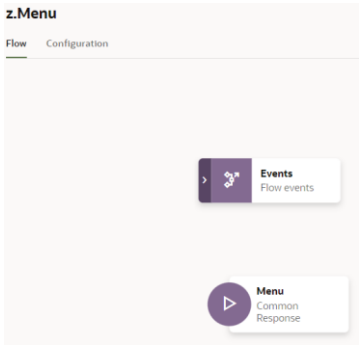
Modify the Menu

Now that we have the questions that will be covered by our bot, we can help the user understand how to interact by using the Menu feature included in this template.

If you type **Menu** in the tester, you will see that the bot is able to show some buttons with different topics that are covered. By default, this template covers topics around the Oracle Cloud. We are now going to customize these buttons.



1. Click **Flows** in the left navbar to open the Flow editor. The editor opens to the Main Flow, where you set and manage the configuration for the entire skill.
2. The flow we have to modify is called **z.Menu**. **Click** on it and the visual flow designer will open.



There's one main state in the flow: Menu state: here is where the menu is defined.

3. Click on the **Menu** state.
4. Click on **Edit Response Items** in the new tab that has opened.

```
responseItems:
  - text: "${rb('z.MenuOptions')}"
    type: text
    actions:
      - payload:
          variables:
            system.text: "${rb('systemFlowName_dummy.ans.intent1')}"
          action: system.textReceived
        label: "${rb('systemFlowName_dummy.ans.intent1')}"
        type: postback
      - payload:
          variables:
            system.text: "${rb('systemFlowName_dummy.ans.intent2')}"
          action: system.textReceived
        label: "${rb('systemFlowName_dummy.ans.intent2')}"
        type: postback
      - payload:
          variables:
            system.text: "${rb('systemFlowName_dummy.ans.intent3')}"
          action: system.textReceived
        label: "${rb('systemFlowName_dummy.ans.intent3')}"
        type: postback
```

5. **Modify the code** with the text you'd like to show in the buttons. The parts that need to be modified are the properties "`${rb('systemFlowName_fullintentname')}`" that refer to the intents you want to have in the menu.
6. Click on **Apply**.

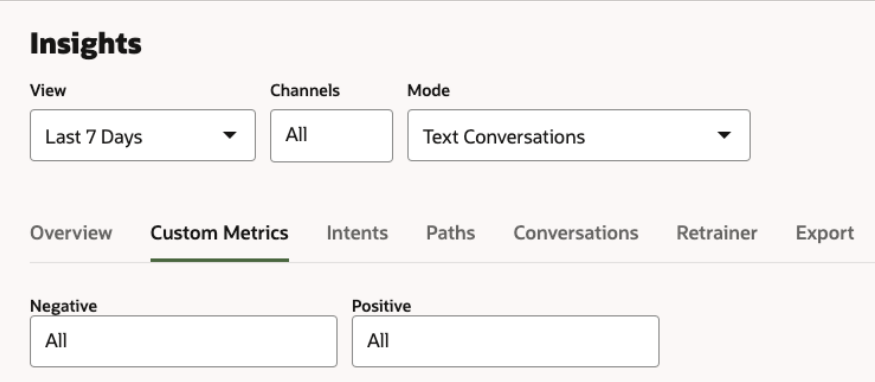
Let's try now the new Menu. Click on Preview and type "Menu" in the chat. You will be able to see the Menu you've just created.

Feedback feature

The template includes a way of collection feedback by showing 👍 and 👎 after an answer is being shown. The user can let the bot know if the information is what she was looking for or not.

As a owner of the bot, we can see this information by doing this:

- **Click** in the left navbar to access the insights feature.
- Click on **Custom Metrics**
- Negative and Positive reactions will be shown to all questions by default



- You can select specific intents by clicking in the field below Negative and Positive.

Understand the Insights Module

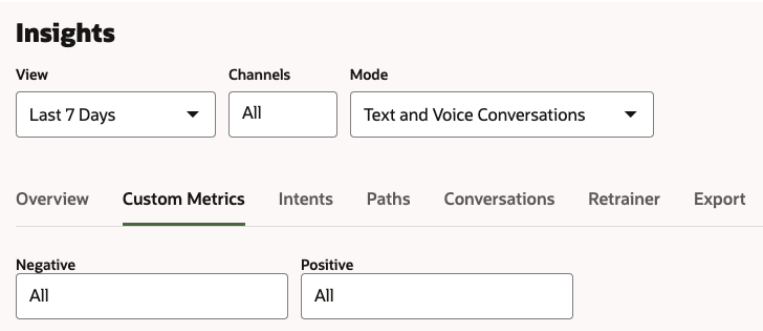
The Insights reports offer developer-oriented analytics that pinpoint issues with skills. Using these reports, you can address these issues before they cause problems.

You can track metrics at both the chat session (or user session) level and at the conversation level. A chat session begins when a user contacts a skill and ends either when a user has closed the chat window or after the chat session has timed out after a period of inactivity. A chat session can contain multiple conversations. You can toggle between the conversation and session reporting using the Metric filter in the Overview report.

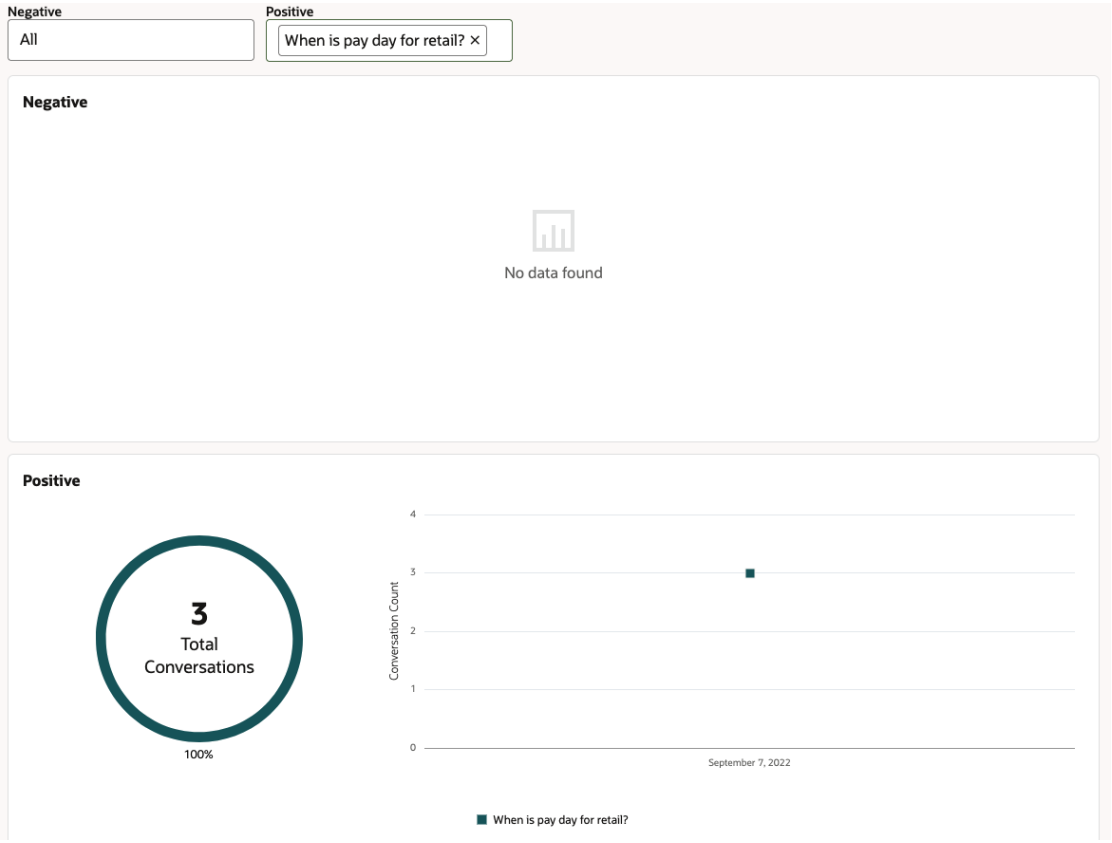
In this training we are going to cover two important points for this module: feedback review and managing unresolved utterances.

Feedback Module

1. **Click** in the left navbar to access the Insights module.
2. Select **Custom Metrics** in the menu.



3. By default, you'll be able to see full metrics, with data for all the intents in your skill. If you want to see an specific intent, negative or positive, type in one of the fields the name of the intent and select it. Leave the other field to All.



Unresolved Utterances

1. Within the Insights section, click on **Retrainer**.
2. Under the Results section you'll see the utterances that matched with the Unresolved Intent. This means that these utterances weren't recognised by the chatbot and no answer was provided. For this, we can do two things:
    a. Check if any of the utterances matches with one of the questions we have defined and attach it to the intent.
    b. Create a new answer intent out of questions that have been made by the user if we believe this should be included in the bot.

Utterances that match with an existing intent

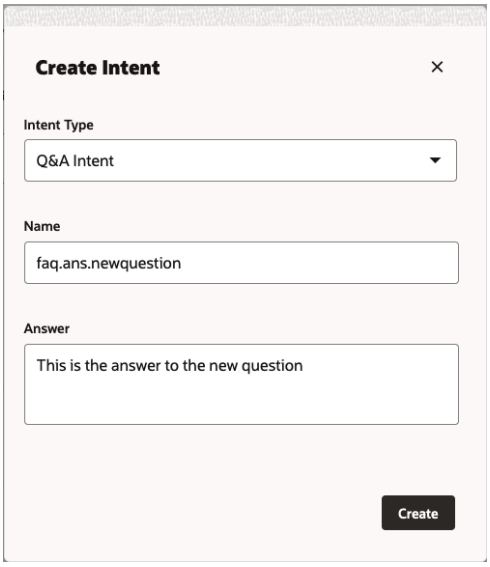
To add the utterance to an existing intent:

1. **Select** the utterance.
2. In the first drop down list, select the **intent** you want to link it to.
3. In the second drop down list, select the **language** of the utterance.
4. Click **Add Example**.

Create a new Answer Intent out of unresolved utterances

1. Select the utterance.
2. In the first drop down list, go to the last item and select **Create Intent**.
3. In the new pop up window, **select Q&A intent**.
4. Add the **name** for the intent. (Use same prefix again e.g. **faq.ans.**) The conversation name (e.g.: When is pay day?) will be added later.

5. Add the **answer**.



6. Click **Create**.
7. Go to the **Intents** section and you'll find there your recently added question. You'd need to add the utterances and a Conversation Name. Please, refer to section **Adding them manually** to refresh how to do it.

Conditional Answers

There are times that we need to route the request depending on different conditions. There are several ways, but as starting point we are going to show you in this tutorial the most simple one.

Let's first create the intent that will trigger this menu. The creation is very similar to the answer intent creation we followed before:

1. Click **Intents** in the left navbar.
2. Click **Add Intent**.
3. Click to enter a descriptive name or phrase for the intent in the **Conversation Name** field.
4. Add the intent name in the **Name** field. If you don't enter a conversation name, then the **Name** field value is used instead.
5. In the **Examples** section, add training utterances that reflect typical ways that users would express the question that the intent is answering. For this example, let's use these ones:
    - What are the opening times?
    - When is the store open?
    - When do you open?

    These 3 utterances should be enough for our example, but keep in mind that for production chatbot we need to include more.

Next we need to create the flow associated to this intent.

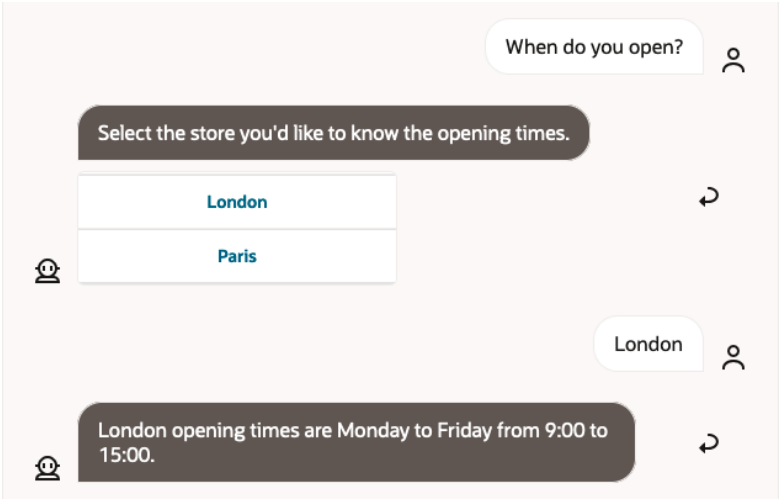
1. Click **Flows** in the left navbar to open the Flow editor.
2. Click **+ Add Flow**.
3. Enter the information for the new flow and select the **Intent Name** of the intent we've just created.
4. Click **Create**.

5. The Visual Flow Designer for the flow is shown now. **Hover over** the Start module and click on the **...** to **Add Start State**.
6. We need to select the **Create Action Menu** template. You can directly search for it in the search field or select User Messaging – Create Text Menu – Create Action Menu. Provide a name and click **Insert**.
7. A new tab opens in the right hand side. Click on Edit Response Items and copy this code:

```
responseItems:
  - text: "Select the store you'd like to know the opening times."
    type: text
    actions:
      - payload:
          action: london
        label: London
        type: postback
      - payload:
          action: paris
        label: Paris
        type: postback
```

IMPORTANT: Spaces are critical in this code. Make sure they look exactly the same as in here when you copy it to the console.

8. To define what to do whenever the user selects one of the option, we need to add transitions. Click on Transitions and click on the + sign next to Action.
9. We need to add as many actions as we have defined in out code. In this example, we have two actions: `london` and `paris`. When selecting an option for **Transition** To, select **Add State...**
10. For the new state, we are going to use the **Send Message** template (first on the list). Select it, give it a name and click Insert.
11. Back in the flow, select the state we've just created and go to **Component** in the menu on the right hand side.
12. Fill the **Messages** field with the information for the store associated with the action. For example: `London opening times are Monday to Friday from 9:00 to 15:00`.
13. Once you've done this for all the actions, click on **Train** (be sure that Trainer Tm is selected).
14. When the training is done, click on **Preview**.
15. Enter a question in the Tester: when do you open?
16. The menu will be shown. Select one of the options and you'll get the conditional answer.



Other parameters defined in the template

The skill template defines 3 skill parameters. These can be modified in the settings tab (last icon in the left navbar), and then click on Configuration. You'll see them at the bottom of the page.

- **answerIdentifyingPattern** - The skill parameter holds the naming convention you use to identify an answer intent. In the sample, the answer intent naming convention is '**.ans.**', which also is set as the default. If you want to use a different naming convention, then change this parameter value accordingly
- **da.answerIntentMinimumThreshold** - To resolve and display an answer, the model uses the confidence threshold defined on the skill settings, which usually is set to 0.7 (Out of 0 - 1). To find related answers you want to define a confidence threshold that must be met by an answer intent for a given question. In the image below, this value us set to 0.3. Note that the lower the value is the more likely it is that the answer is not really related to the top-scoring answer.
- **da.numberOfAnswersDisplayed** - This parameter defines the number of cards and thus the number of related answers (if found) displayed to the user. If, for example, 10 runner-up answers resolve above the value set for the **da.answerIntentMinimumThreshold** parameter, then the top-n answers are displayed, where top-n is the value specified in this parameter.

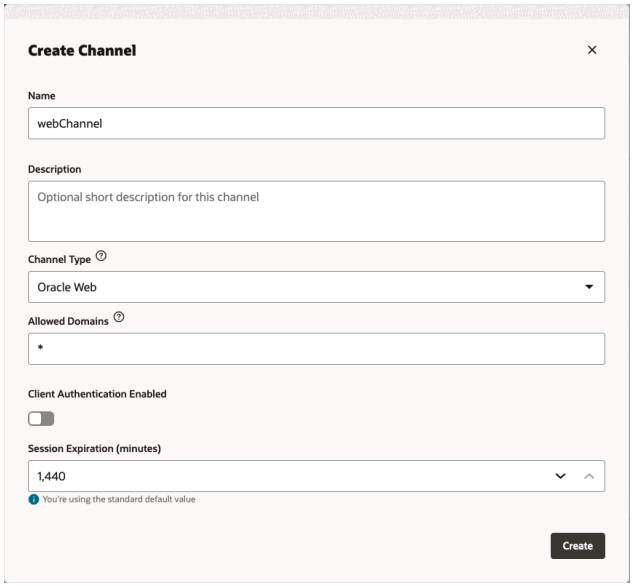
Using the Web SDK

We've been testing the concierge template within the tester in the Oracle Digital Assistant Console, but a chatbot needs to be exposed to be accessible. This can be done to different channels: web, WhatsApp, Twitter…

We're providing a web template for you to deploy the bot and make it public, so other users can test it. Let's see how to make it work.

Create the channel

1. Click on the hamburger menu, select **Development** and then **Channels**.
2. Click **+ Add Channel**.
3. Provide the following fields:
    a. Name: any name to your new channel.
    b. Channel type: Oracle Web
    c. Allowed domains: *
    d. Client Authentication Enabled: Disabled



4. Click **Create**.

5. The channel configuration page opens automatically. First thing you need to do is to route your channel to your skill. Select your skill in the **Route To** field.
6. Enable the channel with the toggle for **Channel Enabled**.
7. You will need the **Channel ID** to configure the Web SDK. Copy it in the clipboard. It looks similar to this: `b4ecabba-9fa4-ec4d-9932-9a7d4f04a2f9`.

Modify the web SDK

1. Extract the web.zip file with the Web SDK that has been provided to you.
2. Open in a text editor the **concierge.js** file found in the folder scripts.
3. Modify lines 3 and 5:
    a. Line 3: replace the URI with the host part (without the https://) of your ODA Console URL.
    b. Line 5: replace the channelID with the Channel ID you copied few steps before.
4. **Save** the file.

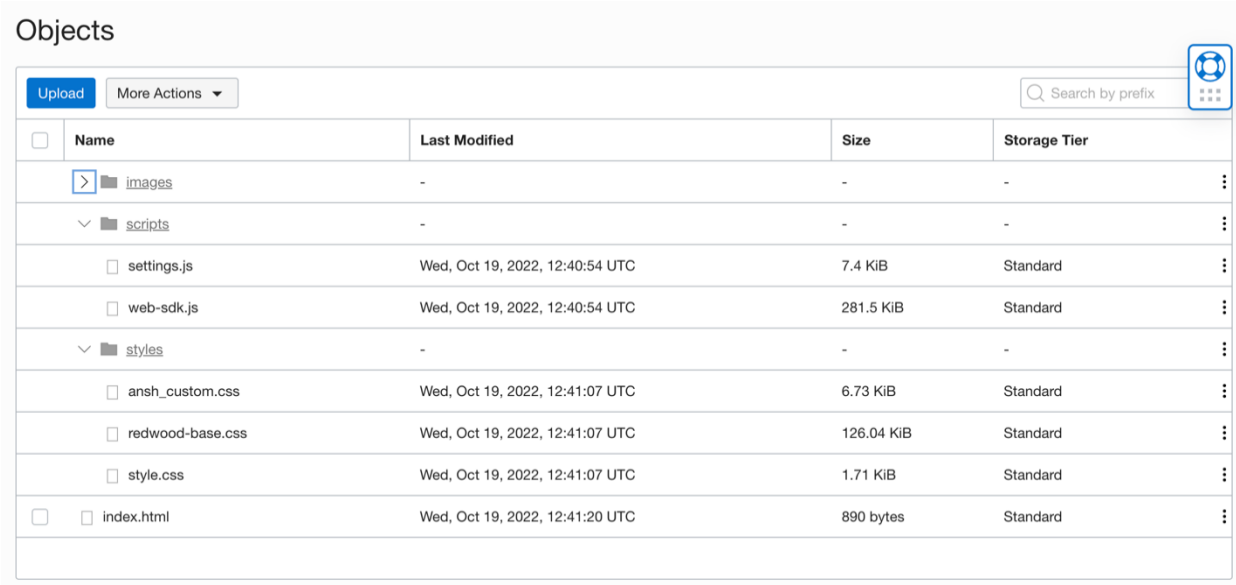
Now open the index.html file in the root folder for the web SDK. You'll be able to test your skill from there. If you want to expose it as a public website, we suggest to use Public Buckets in Oracle Cloud. Find more information here.

Hosting your Web SDK through Object Storage

1. Open cloud.oracle.com and **search for buckets** or navigate to Storage / Object Storage & Archive Storage / Buckets.
2. **Select the compartment** where you want to create the object storage on the left hand side and then press Create Bucket button
3. Give the name of the bucket you want to. Let us say ODA_FAQ in this case and press Create
4. Once the bucket is created, change the visibility of the bucket and make it to **public** so that anyone can access the link.
    a. To do so open the created bucket and press **Edit Visibility button**
    b. **Change it to Public** and press Save Changes
5. Now that our object storage is ready, we need to upload the files inside the web folder in your desktop here. To do so we first need to create the folders in the same name as we have inside the web folder (images, scripts, styles). To create the folders Press on More Actions → Create New Folder

    Create all 3 folders as said earlier
6. Once the folders are created we need to upload the files from the web folder in our desktop. To do so press the upload button inside each and every folder and upload the files respectively.

    Once the files are uploaded to their respective folders. The folder structure will look like this



6. Finally , we just need to get the public url of the index.html file and share it with whomever we want to. To get the url Click on the three dots on the right hand corner of index.html file and press View Object Details and copy the url path.