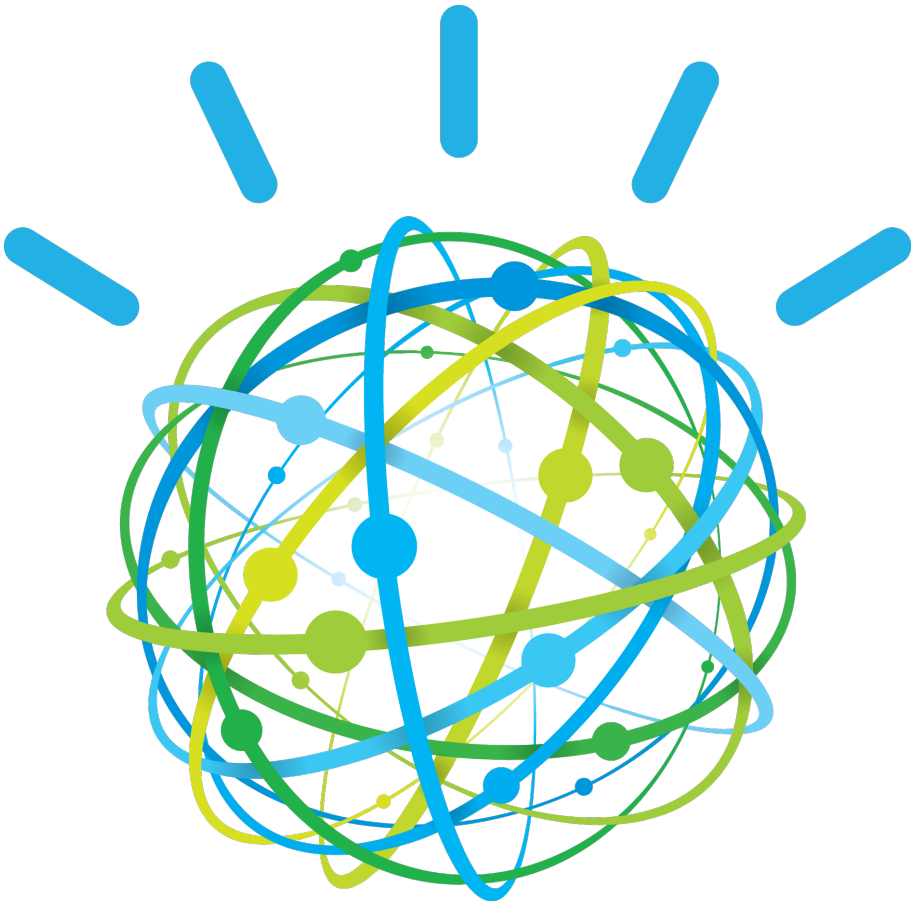
Lab 1: Building a conversation

IBM Watson Assistant



Lab Instructions

Laurent Vincent

Content

Let's get started

1. Overview

The IBM Watson Developer Cloud (WDC) offers a variety of services for developing cognitive applications. Each Watson service provides a Representational State Transfer (REST) Application Programming Interface (API) for interacting with the service. Some services, such as the Speech to Text service, provide additional interfaces.

The Watson Assistant (WA)service combines several cognitive techniques to help you build and train a bot - defining intents and entities and crafting dialog to simulate conversation. The system can then be further refined with supplementary technologies to make the system more human-like or to give it a higher chance of returning the right answer. Watson Assistant allows you to deploy a range of bots via many channels, from simple, narrowly focused bots to much more sophisticated, full-blown virtual agents across mobile devices, messaging platforms like Slack, or even through a physical robot.

The **illustrating screenshots** provided in this lab guide could be slightly different from what you see in the Watson Assistant service interface that you are using. If there are colour or wording differences, it is because there have been updates to the service since the lab guide was created.

2. Objectives

In this lab, you will:

- Learn how to use the IBM Cloud web user interface to create and manage Watson services
- Learn how to train your chat bot to answer some asked questions

3. Prerequisites

Before you start the exercises in this guide, you will need to complete the following prerequisite tasks:

- Download all files needed for this lab using the link provided by the instructor provided you the link to get labs content. You may download each file individually.

4. Scenario

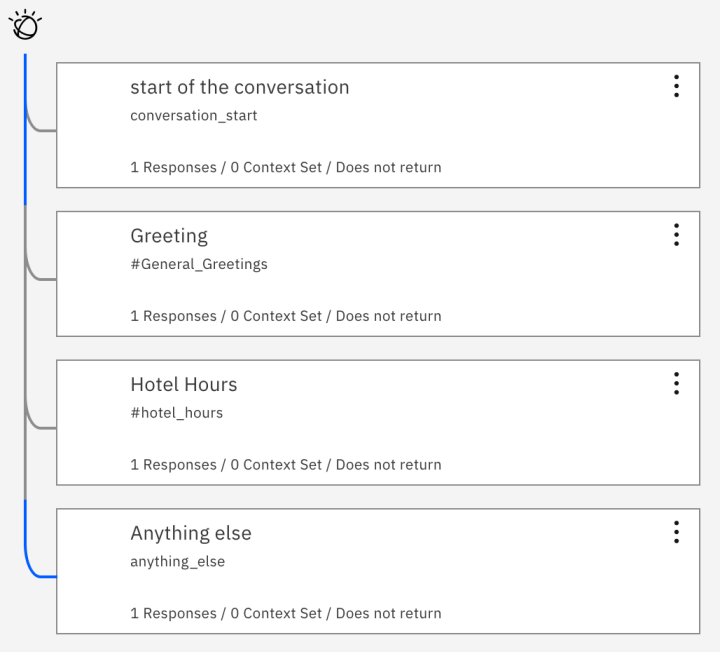
Use case: A Hotel Concierge Virtual assistant that is accessed from the guest room and the hotel lobby.

End-users: Hotel customers

5. What to expect when you are done

At the end of this lab, you should have a simple dialog using

- around 23 intents,
- around 11 entities.



Create an Assistant in IBM Cloud

6. Create your Watson Assistant Service

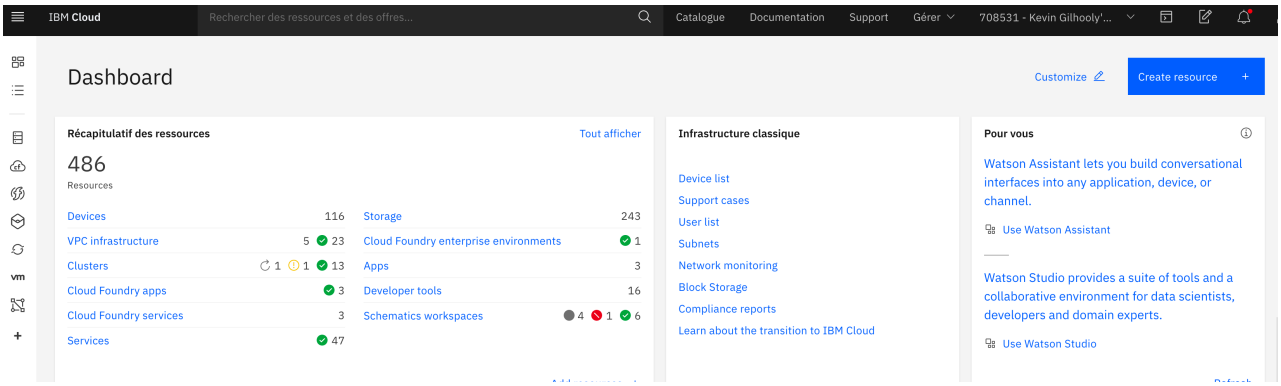
IBM Cloud offers services, that provide additional functionality that is ready to use for your application's running code.
You have two options for working with applications and services in IBM Cloud: via web user interface or the command-line interface. In this lab, you will use the web user interfaces.

1. In a web browser, navigate to the following URL:

Location	URL
worldwide	https://cloud.ibm.com/login

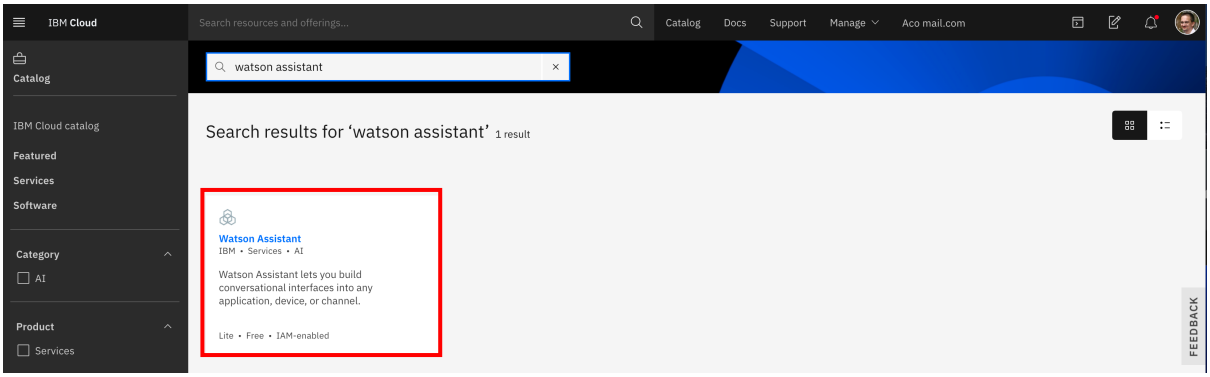
2. Log in with your IBM Cloud credentials. This should be your IBM ID.

You should start on your dashboard that contains metrics related to your IBM Cloud activities.



3. Click on **Create resource** (top right of the page)

4. In the search box, enter *Watson Assistant* as filter



5. Click on the **Watson Assistant** service tile.
6. Review the details for this service. To see a description of the service, open the tab **About**. After you viewed the description, open the tab **Create**, and review the pricing plans. Notice that the Plus trial provides an access to all features for 30 days.
7. At the top, you enter the geographical location for your new service. Select location *Frankfurt .* If Frankfurt is not available, choose another location close to you, such as London.

Note: remember which location you chose! You will use this same location in next labs.

8. In the pricing plan, select *Plus Trial*.
9. At the bottom, fill out the fields as follows.

Field	Value
Service name	*Conversation-**XXXXXX**-**LV***
Resource group	default

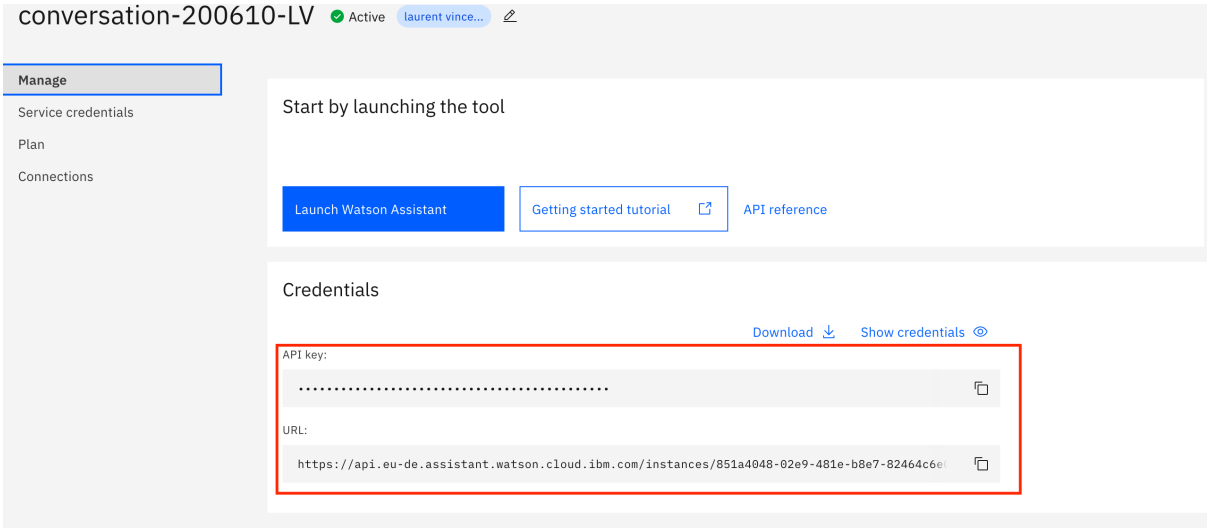
Replace **XXXXXX** with the date of the creation with the format yymmdd.

Replace **LV** with your initials.

10. Then click **Create** (bottom right).

Note: You can provision only one Lite or Plus Trial instance of a Watson Assistant service at a time. So, if you already have such instance, delete it now and create a new one as shown in previous steps.

IBM Cloud will create a new service instance. In order to use this specific instance in your application, you will need to obtain its credentials. The credentials are displayed in the **Manage** tab or in **Service credentials** tab. The credentials consist of **API Key** and **URL** for your service instance.



If you want to view the API key, click on **Show Credentials**. In later exercises, you will enter these credentials into a JSON configuration file for your application. Copy now the credentials to a temporary file, or just return to this IBM Cloud page later when you will need the credentials.

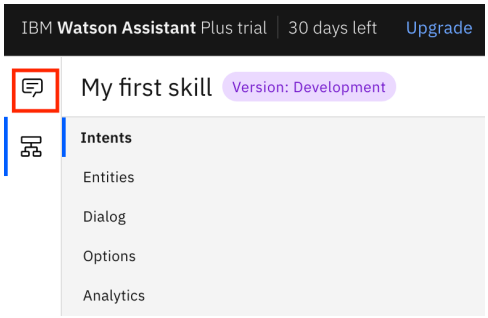
11. Click the **Launch Watson Assistant** button.

Your first step in the Watson Assistant tool will be to create an Assistant and a Skill.

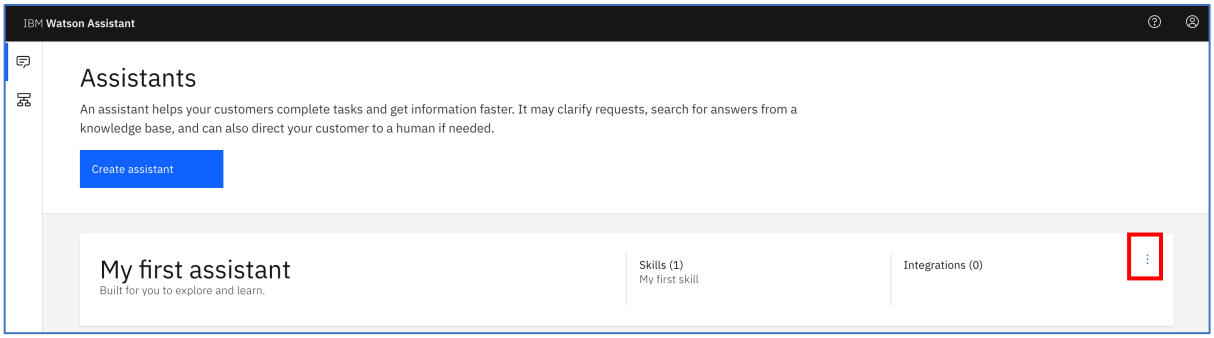
7. Manage your first Assistant

An Assistant is a virtual container to which you add your skills. Assistant enables a skill for interaction with your customers.
The Watson Assistant tool has created for you your first Assistant, your first Skill and named them **My first assistant** and **My first skill**.

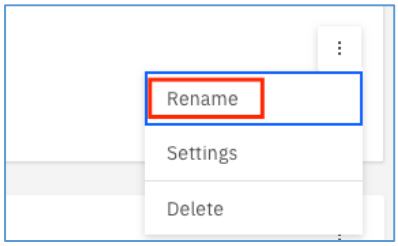
1. You should get a start tour invitation window. Just click outside the window to close it.
2. Rename your Assistant to **Concierge** by following these steps:
    a. Go to **Assistants** page, by clicking **Assistants** icon top left.



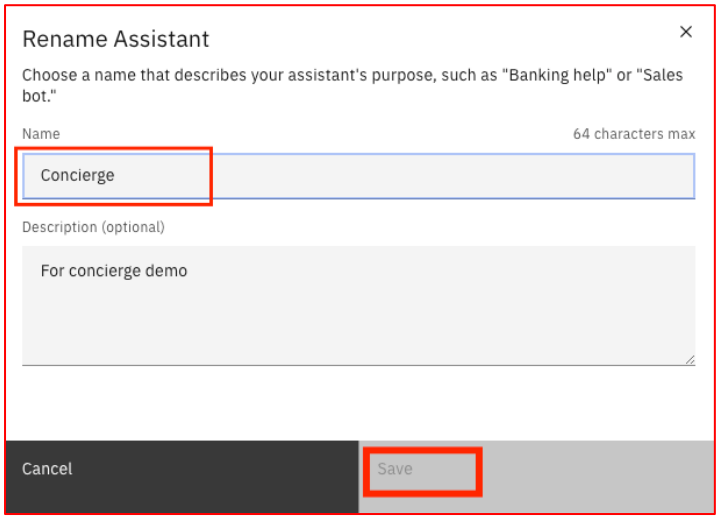
b. In the **My first assistant** tile, click on 3 dots on the right-hand side.



c. Select **Rename**.

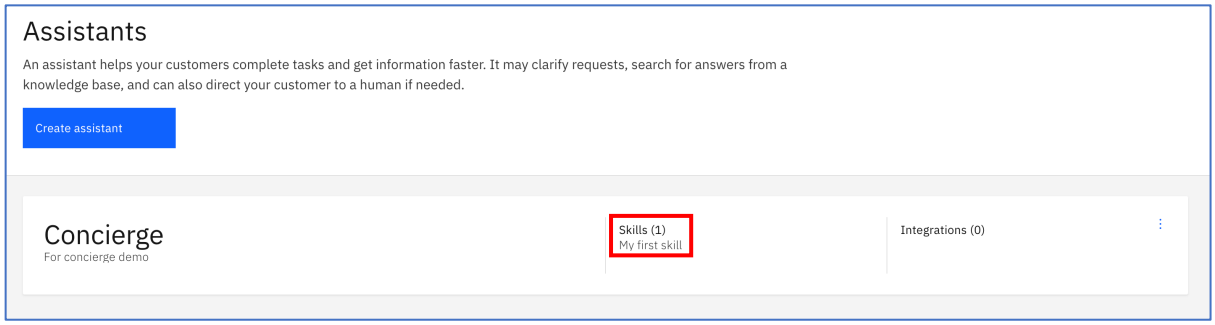


d. Rename your Assistant to *Concierge* and give it a description as shown below.

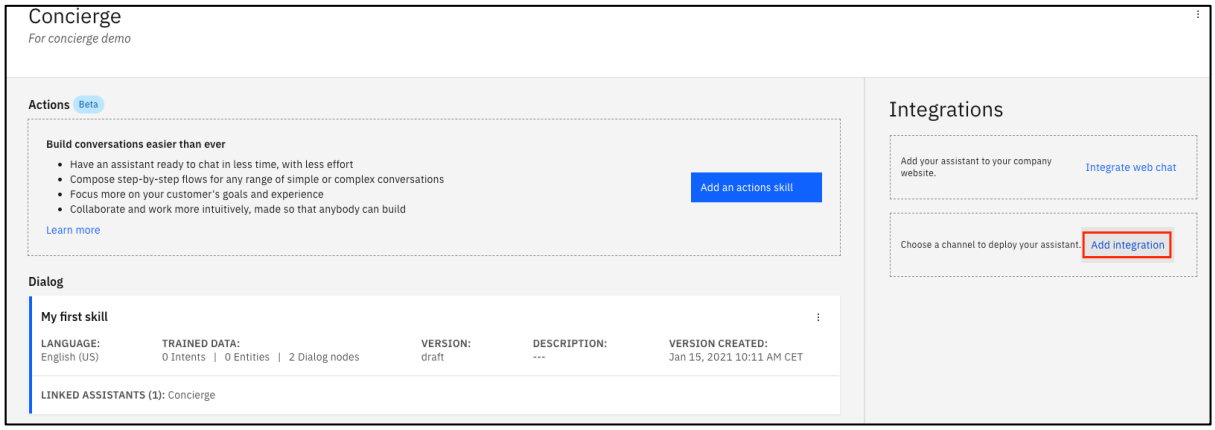


a. Close **Save**.

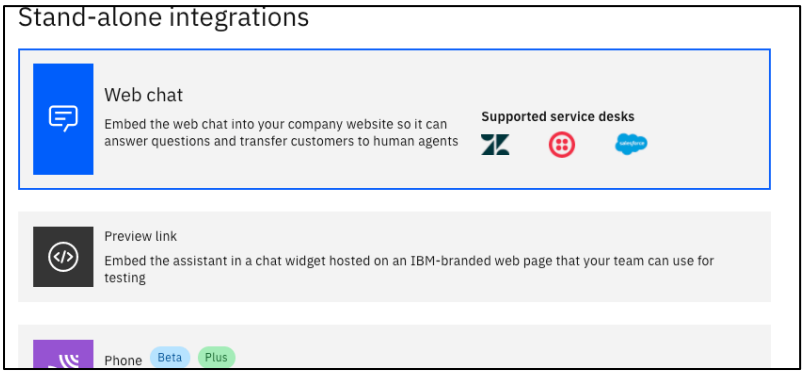
You should see your Assistant with the new name: *Concierge*. You can also see that your Assistant has one Skill which is named *My first skill*. You are going to update the name of the skill later on.



3. Click on the **Concierge** tile
4. (Top right) Click **Add integration**



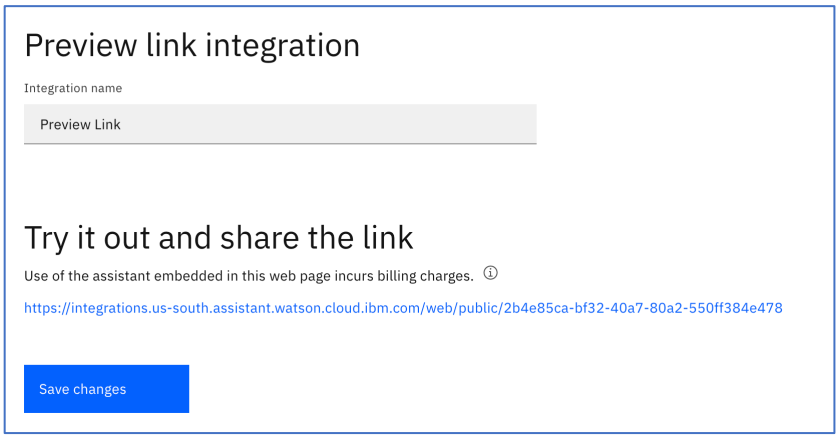
5. Here you can pick several different integration types for your Assistant, such as Web Chat, Facebook Messenger etc. You will choose **Preview link**, so click on it.



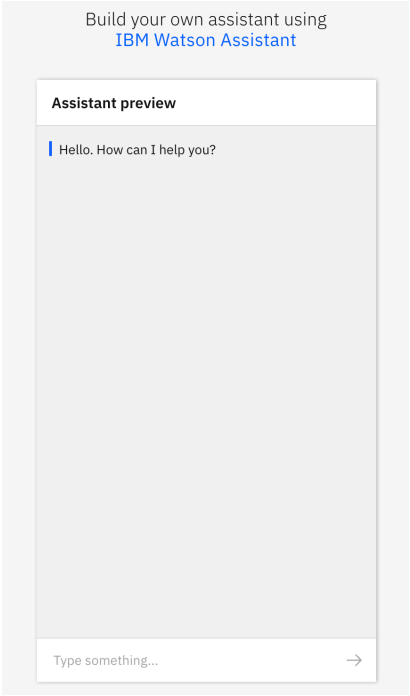
The Preview link is a URL at which your Assistant is available to the outside world.

6. Keep the default setting and click **Create**

7. An URL will be presented to you. This is the external URL (the preview link) of your Assistant.

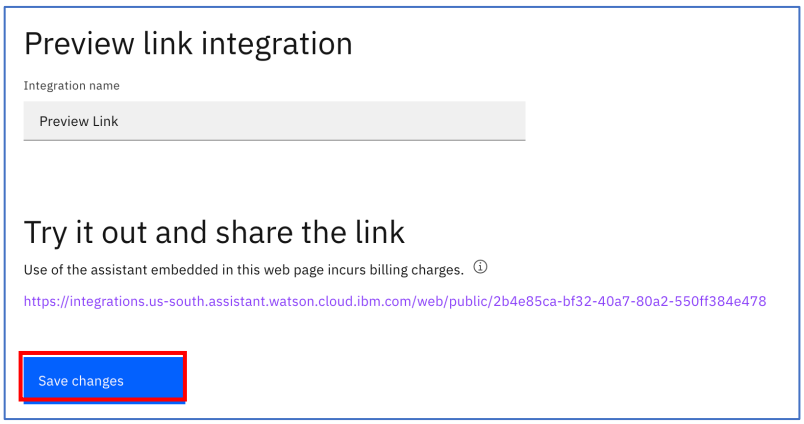


8. Click on the **URL** to see how you Assistant looks to external world. A new tab will open in your browser, looking like this:



9. Your new assistant greets you and waits for your input. **Bookmark** this URL, you will use it later to work with your Assistant as an external user.

10. Go back to the window with the Preview link integration and click **Save changes.**



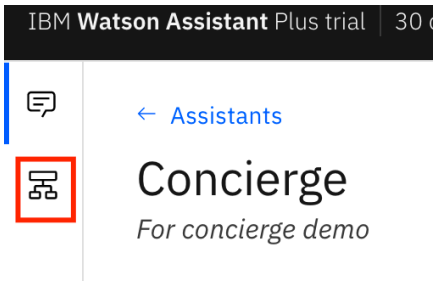
You have successfully managed your first Assistant and created an external URL (preview link).

8. Manage your first Skill

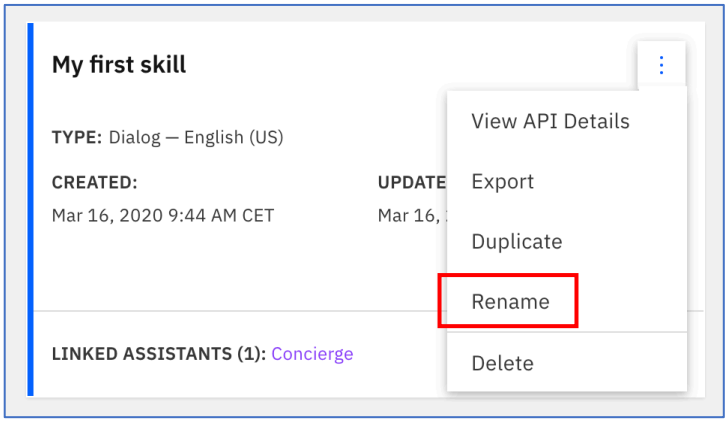
To use the new Assistant, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. You will create these items in a Skill. A Skill is a container for the artefacts that define a piece of the behaviour of your service instance.

You are going to update the pre-created Skill for the lab. For the exercise, you will use the English (US) language which is the default language of a skill. As we cannot update the language of a skill, If you want to use another language (not for the lab as all content is in English), you must delete the default skill and create a new one with the right language.

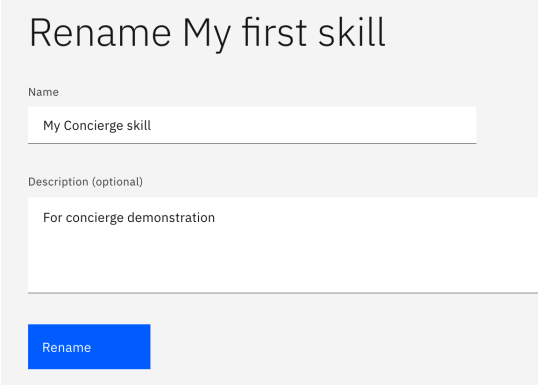
1. In the menu on the left, Click to **skills** icon



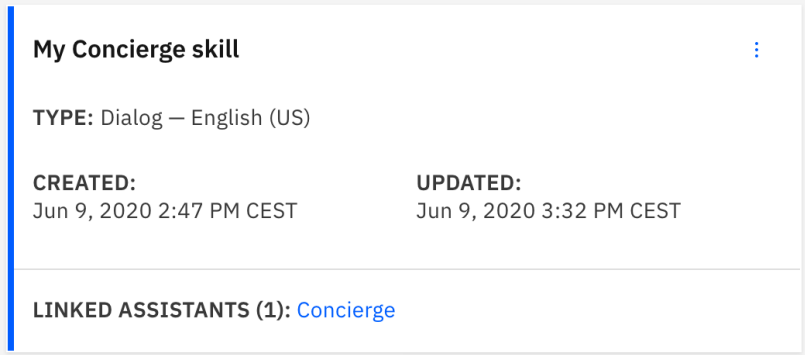
2. In **My first skill** tile, click on three dots on the right, then click **Rename**.



3. Rename your skill to *My Concierge Skill*, give it a description as shown below and click the **Rename** button.



4. You should now see your skill with the new name.



Note: the language of the skill is English (US)

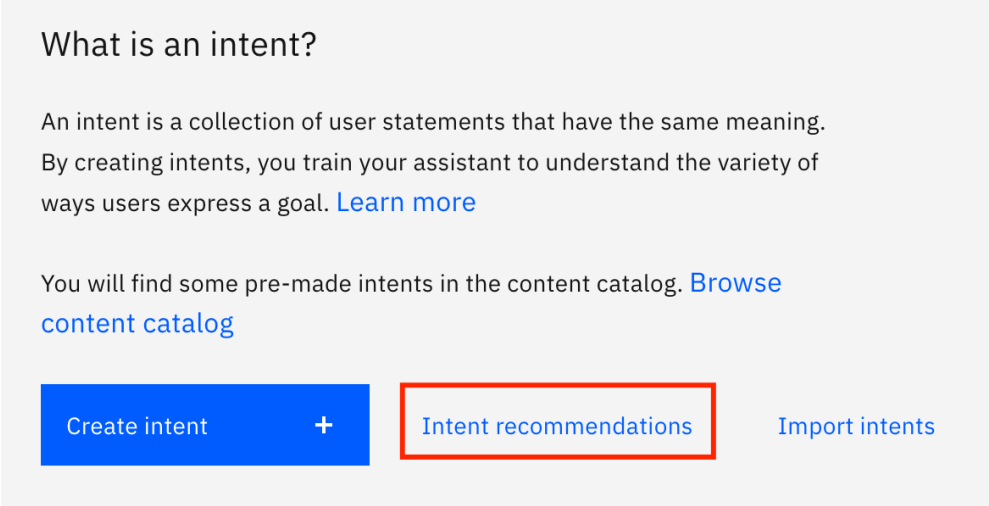
Intents

In order to use the new skill, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. First, you will need to define some intents to help control the flow of your dialog. An *intent* is the purpose or goal of a user's input. In other words, Watson will use natural language processing to recognize the intent in the user's question / statement. This will help Watson select the corresponding dialog branch in your automated conversation.

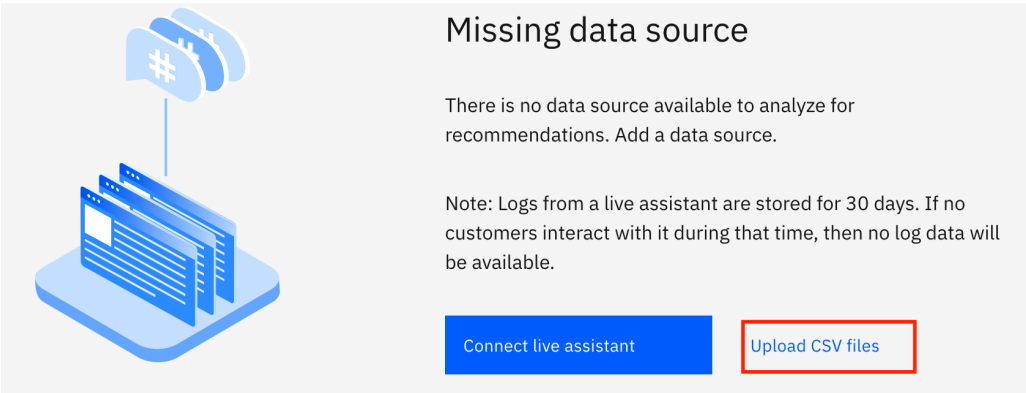
9. Group your user's inputs by intents

The initial activity should be to organise the user's examples you have collected, and group them by intents. To speed up this step we are going to leverage Intent recommendations feature of WA.

1. To open the skill page, click on the tile **My Concierge Skill**
2. If not already there, click the **Intents** tab at the top of your skill.
3. On the **Intents** tab, click **intent recommendations** in the dialog window.

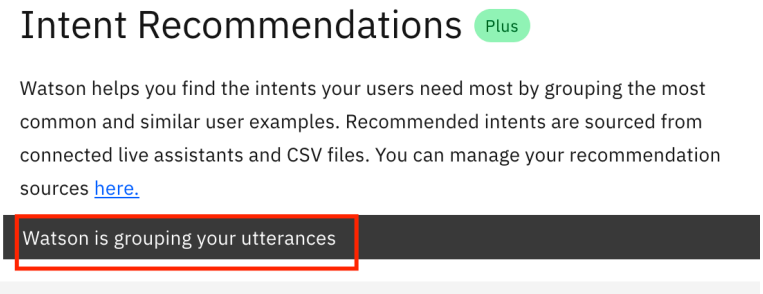


4. The data source is a CSV file, click **upload CSV** in the dialog window.



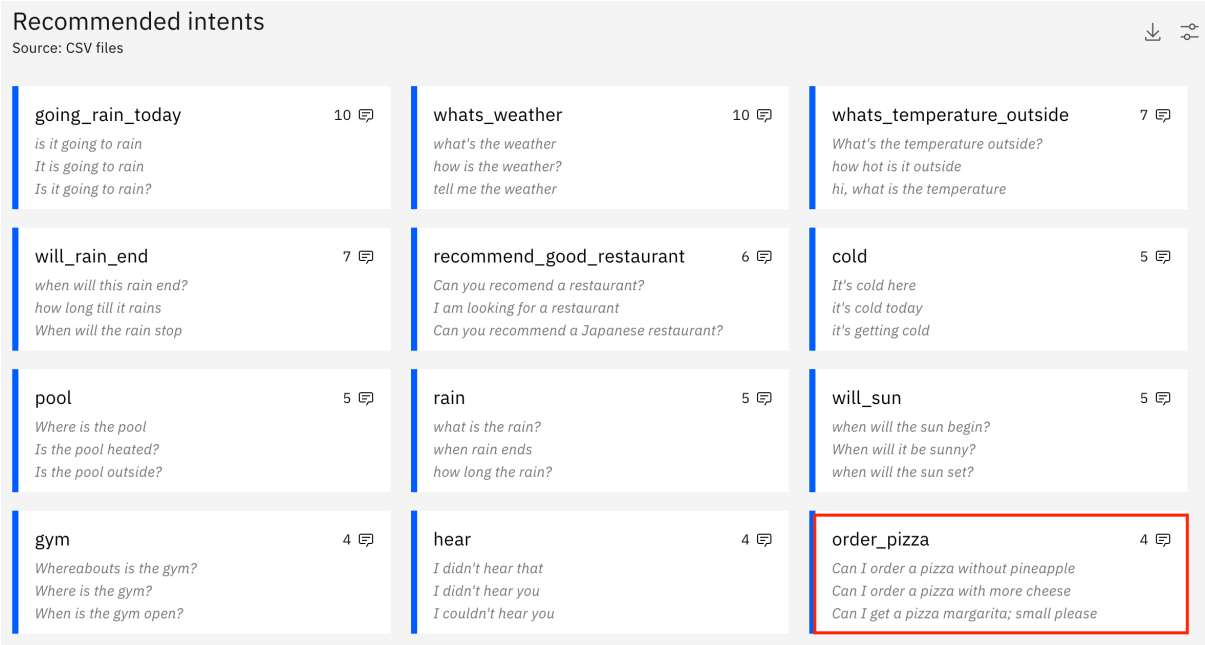
5. Click **Add file** in the dialog window,
   then select the file *WA_Lab_examples Before Session 4.csv* .
   The file is in folder **Labs Sources**. You should have downloaded this folder from the workshop materials before starting this lab. If you haven't already, download it now.

Watson is grouping your utterances.



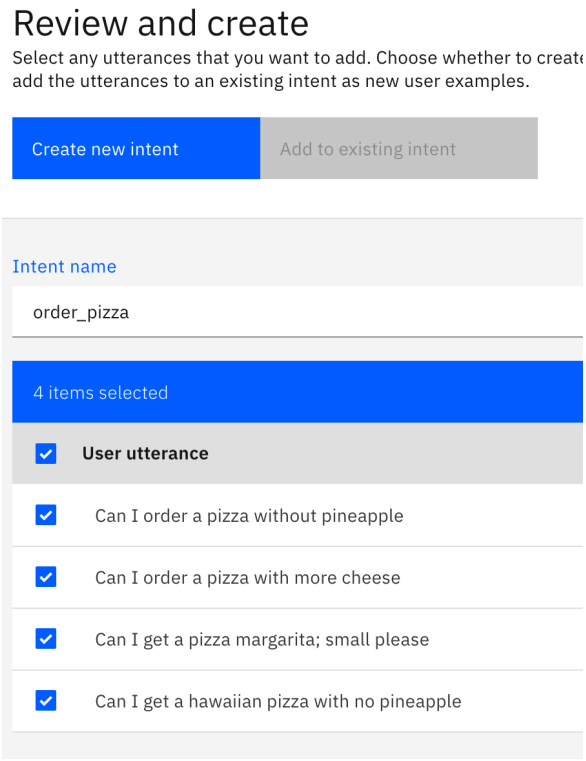
Note: periodically, refresh your page and go back to the intent recommendation to see if you get the result of the job.

After the execution you should have several proposed intents:



6. Click on the tile **order_pizza**

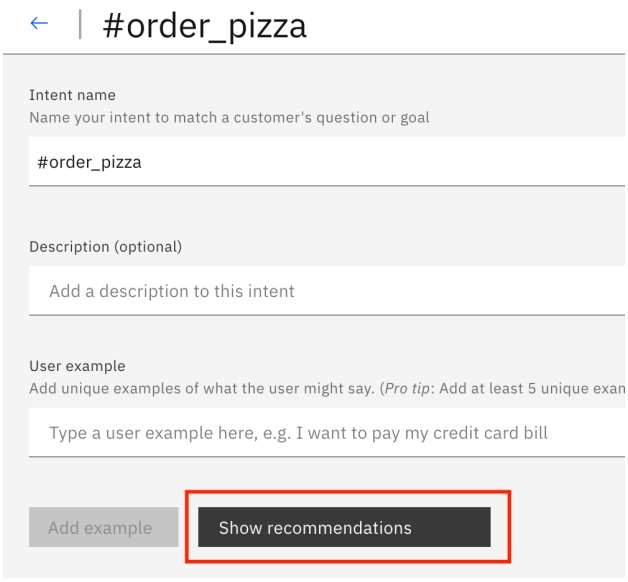
7. Select all relevant examples to order a pizza



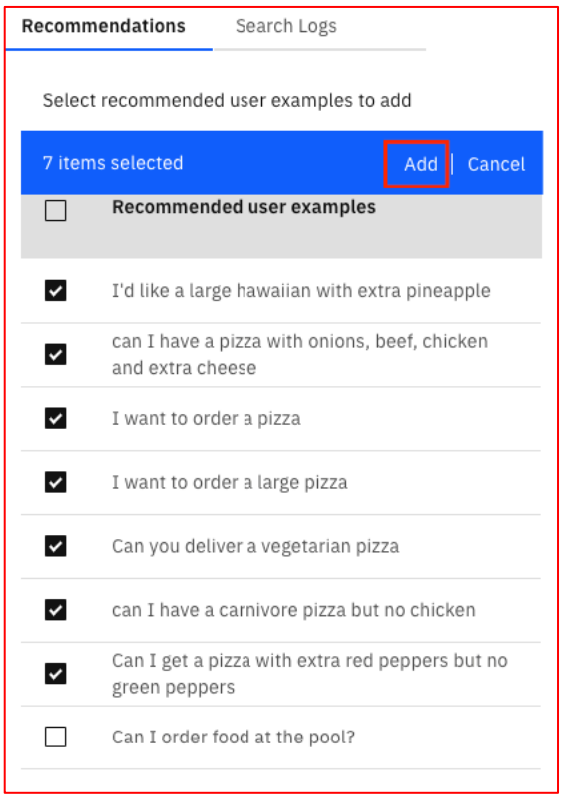
8. Click **Create new intent**

In this case, the Intent *#order_pizza* will be created with 4 examples.

9. We are going to enrich the intent, click **Show recommendations**

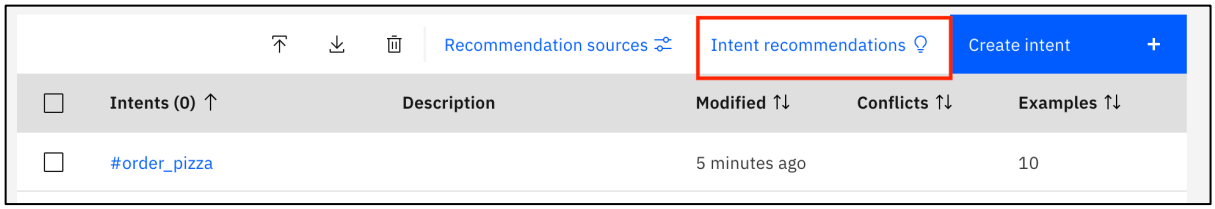


10. On the right frame, select all examples related to “order a pizza”, and click **Add**



11. You could continue to import example form the CSV File but go back to the intent page by clicking on the arrow (top left)

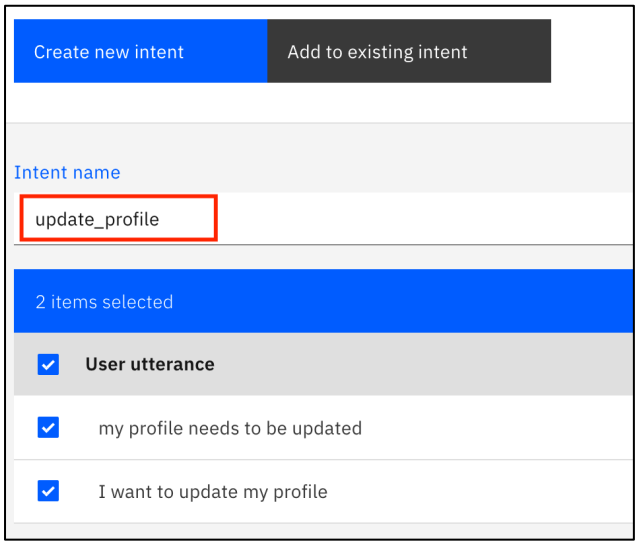
12. Go back to the intent recommendation page by clicking on **intent recommendations**



Note: The order_pizza proposition should have been removed.

13. Scroll down, you should find a tile **profile**, click on it.

14. Select all utterances, update the name with *update_profile* .

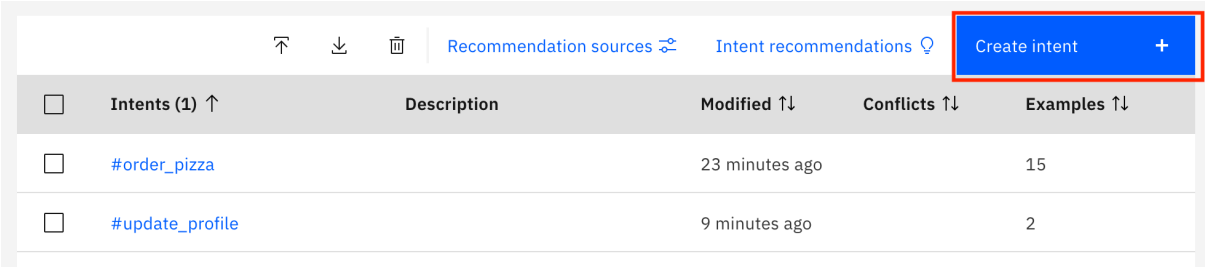


15. Click **create new intent**.

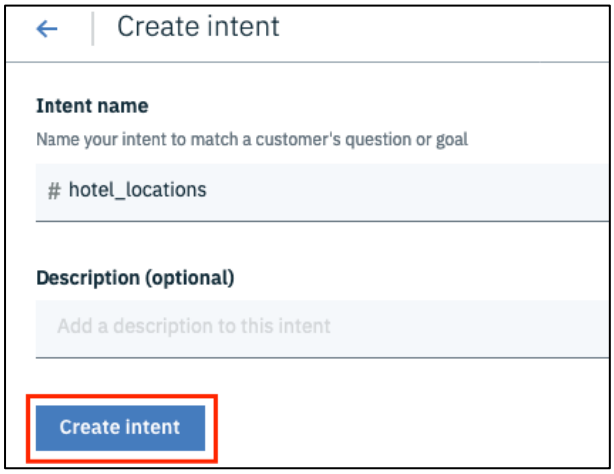
We could continue to analyse the classification proposed by WA. Right now, we are going to use other WA capabilities.

10. Create your first intent

16. If not already there, Go back to the **Intents** tab of your skill.
17. Click **Create intent**.

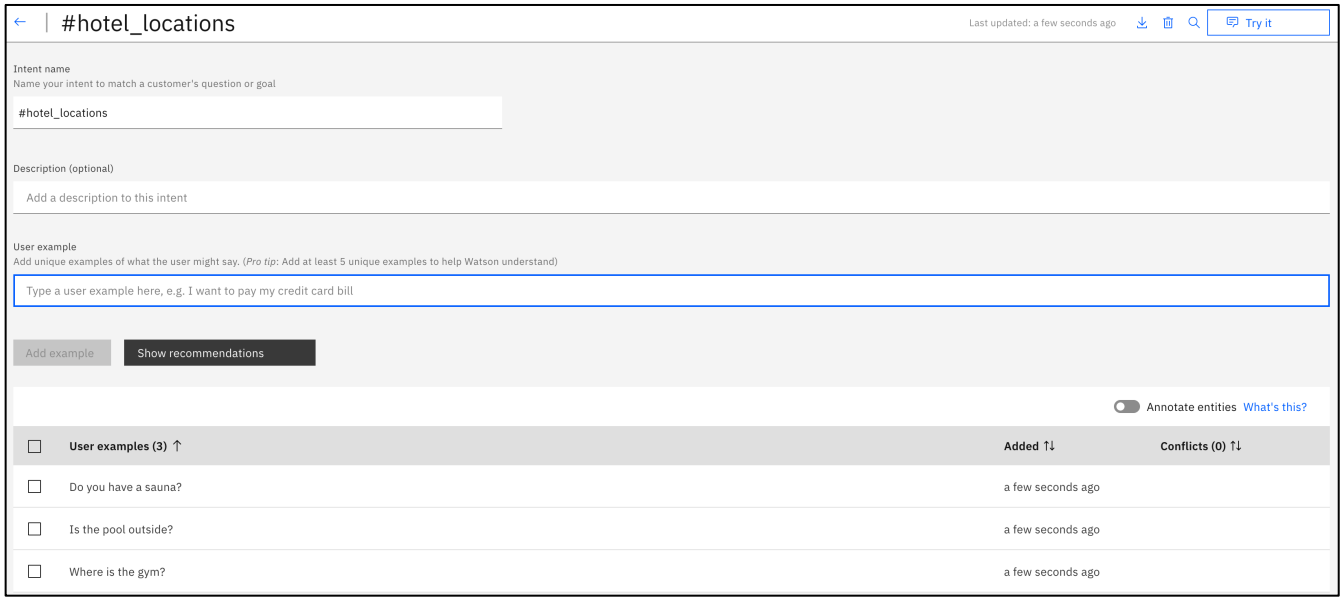


18. Enter *hotel_locations* as Intent name then click **Create intent**



19. Enter the examples as defined below in the **User example** field. Click **Add example** button after entering each example. The *user examples* are phrases that will help Watson recognize the new *intent*.

Field	Value
Intent name	*hotel_locations*
User example	*Is the pool outside?* *Where is the gym?* *Do you have a sauna?*



20. When finished, click on the **arrow** Icon (top left side of the window) to go back to list of existing intents

At this point, you have defined 3 *intents* for the application along with the example utterances that will help train Watson to recognize and control the conversation flow.

11. Import intents

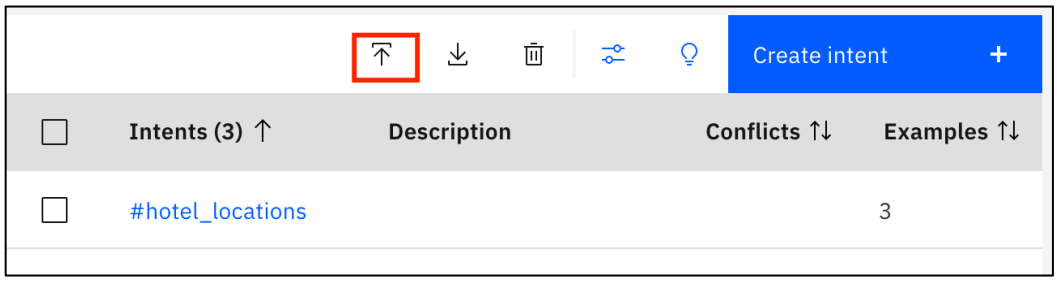
Apart of creating intents manually (as you did in previous step), intents can also be imported into the Watson Assistant tool using a Comma Separated Values (CSV) file saved with UTF-8 encoding.
In order to import the CSV file successfully, the file must include only two columns:

- one column for user examples
- one column for intents.

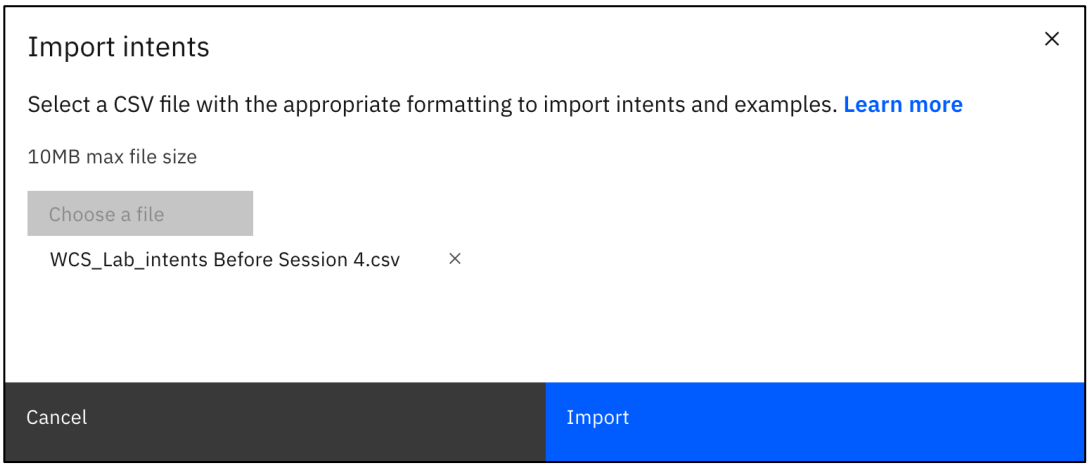
A CSV file that we'll use in this exercise has already been created for you. It was created this way:

- Copy the end-user examples into a single column.
- Label each end-user example with its assigned intent in the column beside it.
- Do not insert any column headings for the intent import file.
- Use appropriate text editor (such as Numbers on Mac and Notepad++ on Windows) to make sure your file is UTF-8 coded and saved as a .csv file. Using the right text editor is important because many text processing software (such as MS Word) add spaces and other hidden characters into a text when it is copied and pasted into a new source. When such file is uploaded into Watson Assistant tooling, spaces might be converted into symbols that will negatively affect learning by adding text that the user did not intend.

1. Click **Import Intents** icon



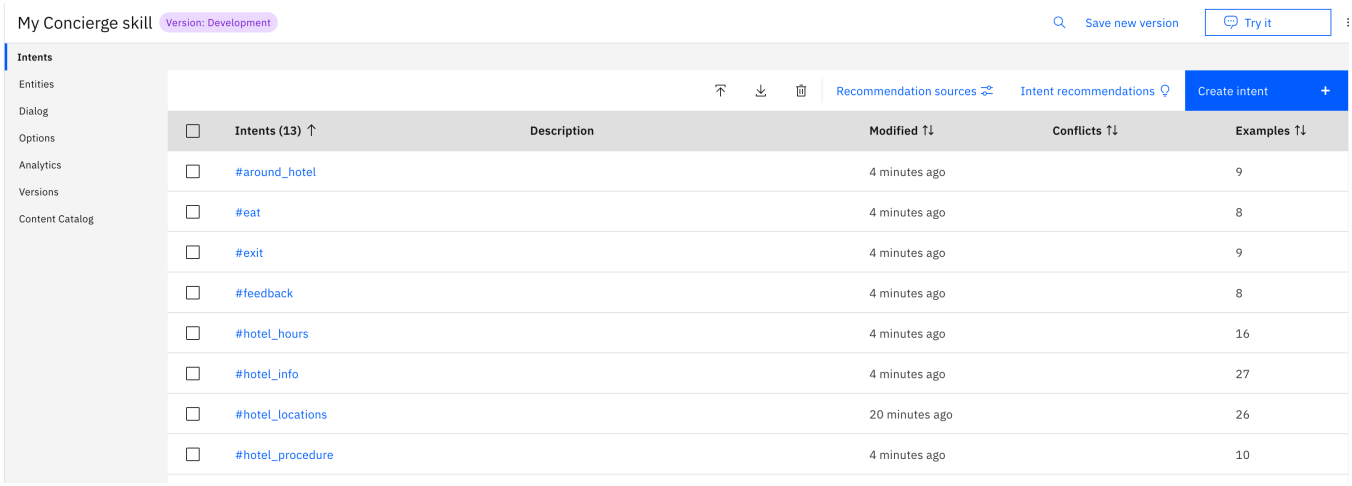
2. Click on **Choose a file** then *WA_Lab_Intents Before Session 4.csv*.



3. Click **Import**

Note: Notice that the Watson Assistant tooling did not import examples that were exact duplicates of questions that you manually entered. Duplicates are not case sensitive. For example, "Do you have a sauna?" is the same as "do you have a sauna". It did import questions that had semantic differences, such as the absence of punctuation, extra spaces between words, or misspelled words.

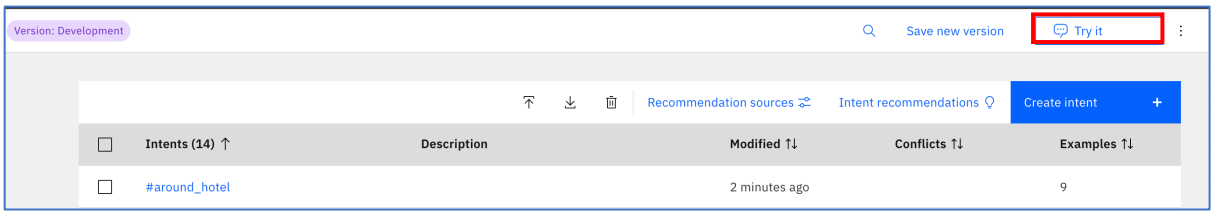
4. Click **Done**, you will see a number of new intents.



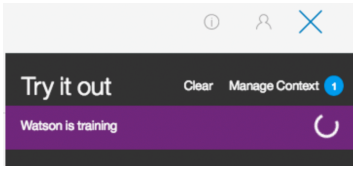
12. Test and improve your Intents

You can use the **Try it** panel for testing your chatbot during initial development.

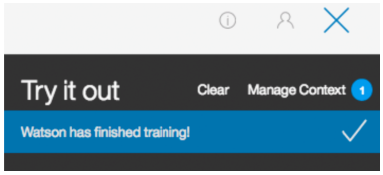
1. Open the **Try it** panel by clicking on its icon (upper right):



When WA is training on the recently added data, you will see a message highlighted with purple: “Watson is training”.

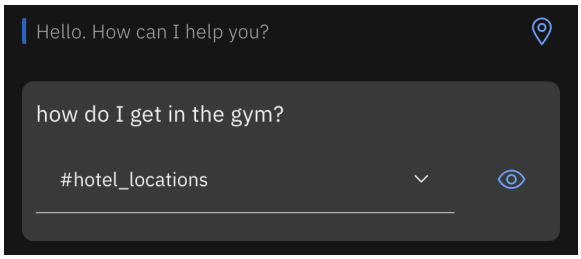


Wait until this message is cleared before you test newly added intents, utterances, entities. Otherwise, Watson will respond, but you will get unpredictable results because the training is not complete yet. When training is finished, the purple box with the text, “Watson is training” will disappear and for a short time you will see a message in blue box saying: "Watson has finished training!"



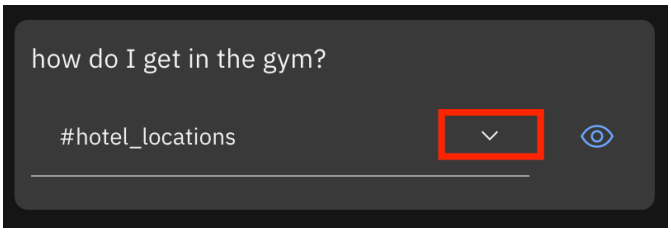
2. In the input field at the bottom of the Try it panel, type in *how do I get in the gym?*

3. Press enter

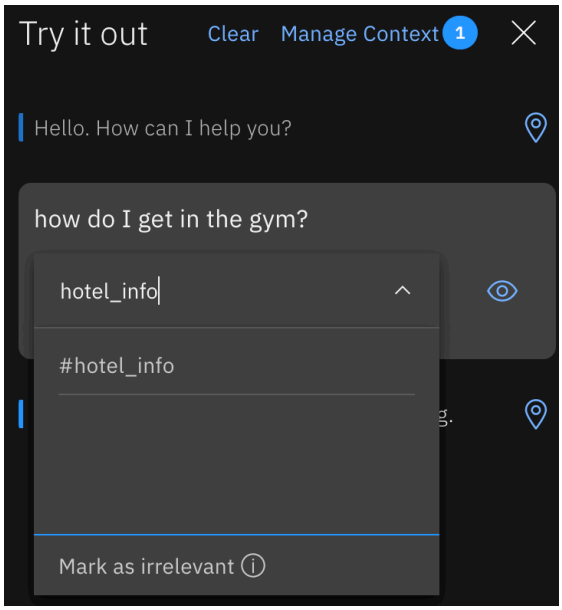


Notice the grey box with the text that you entered. Watson assigned your question to the intent #hotel_locations. The *#hotel_locations* intent is meant for questions that refer to finding directions to places within the hotel. However in this case, the user wants to know how to get in the gym, not how to get to the gym. A small difference in wording can change the correct intent for a particular utterance.
Let's change the intent for this user example to avoid misinterpretation.

4. Click the arrow next to *#hotel_locations*

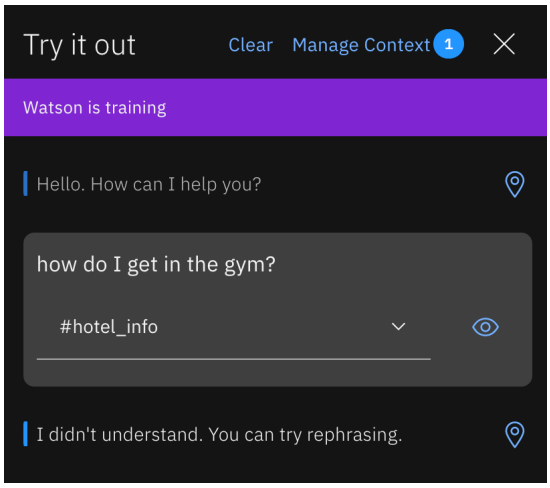


5. Type in *hotel_info*



6. Select the intent *#hotel_info* that appears in the drop-down box

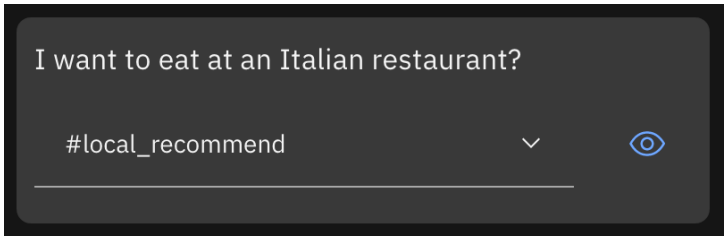
Your **try it out** panel will show a message "*Watson is training*"



Note: If you go to your Intents panel, and open the *#hotel_info* intent, you will now see that your input has been added to this section. As this user's example has been used in the intent *#hotel_locations* this update should cause a conflict between the 2 intents. In another lab you are going to use 'Watson Conflict Resolution' feature to fix it.

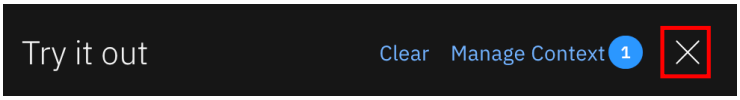
7. Now we will practice changing intent using another technique. Type in Type *I want to eat at an Italian restaurant?*

   Note: Notice that Watson assigns this question the intent *#Local_recommend. (the classification could differ).* This intent is meant for questions that refer to finding recommendation for sightseeing, restaurant in the hotel. In this case, the user wants to eat something.



Let's say we do not like such classification and we want to move our example to another intent. This time, we will do it in a different way - by using the search capability of the engine.

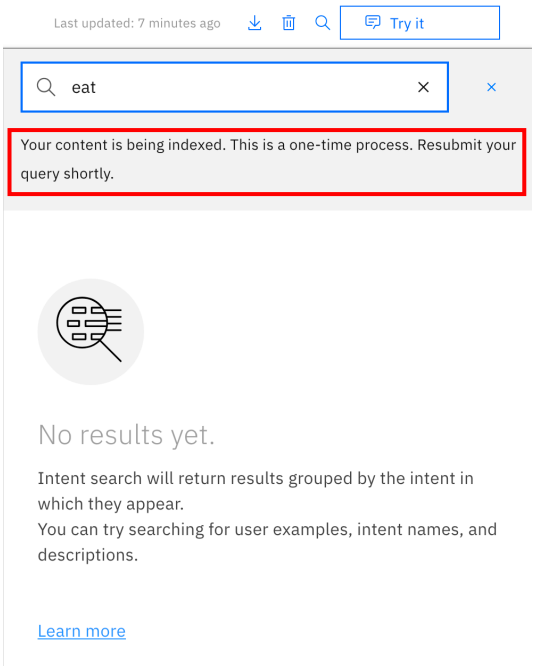
8. Close the **Try it out** Panel by clicking the X on top right.



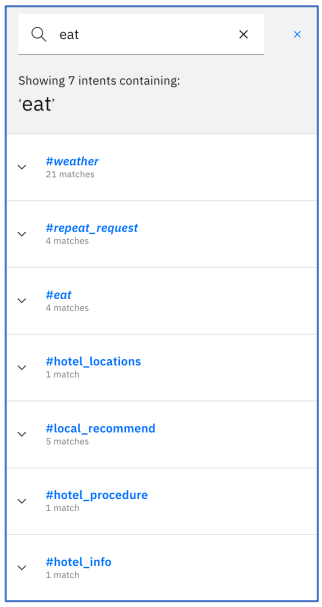
9. Top right, click on the **search** icon

10. Type in *eat* in the search field and press enter.

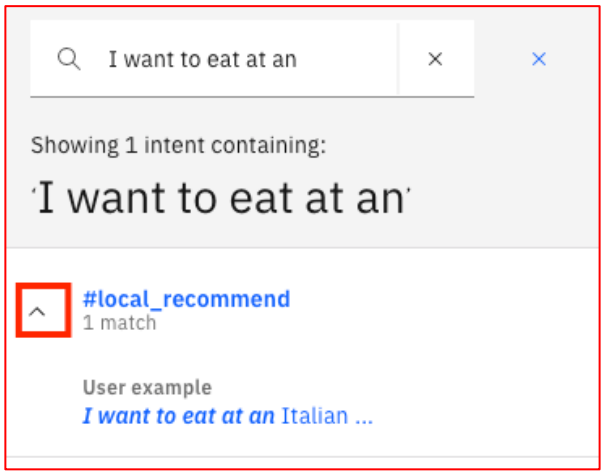
    If you see the message “Your content is being indexed...”, press the refresh button on your browser occasionally until the message disappears. Then resubmit your query.



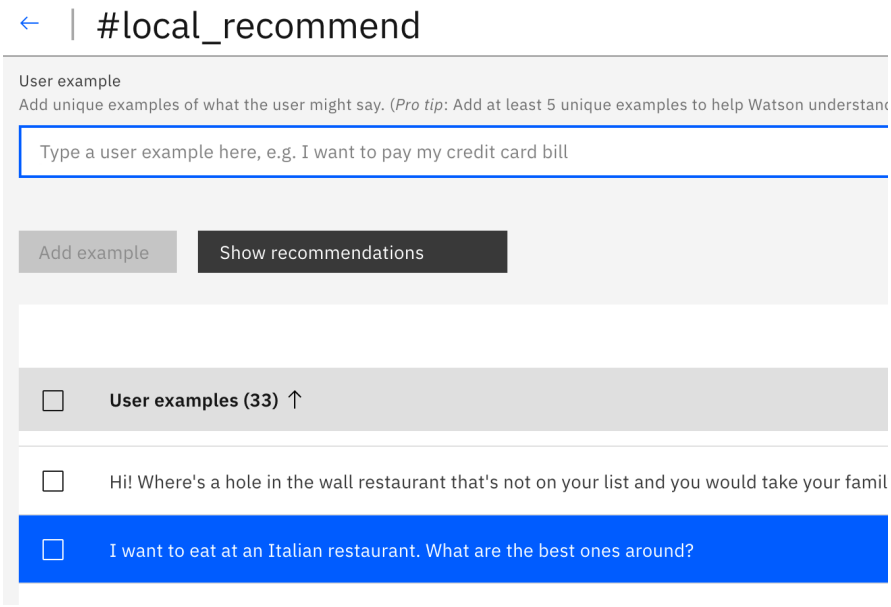
The search returns all intents including the ‘*eat*’ word.



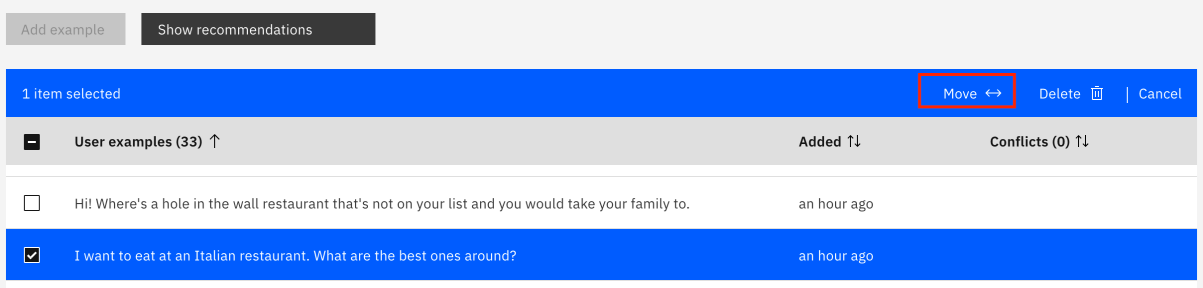
11. Let refine the research, enter for the sentence *I want to eat at* an in the search field, then expand the returned intent *#local_recommend*.



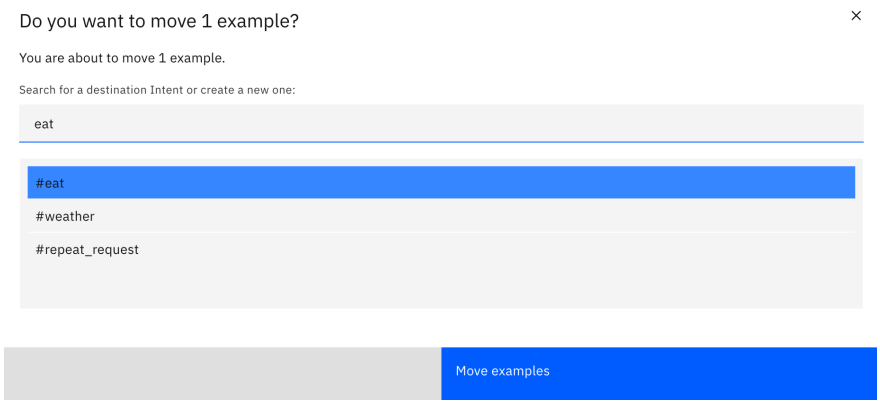
12. We see a list of examples that belongs to this intent (only one in our case). The above example looks like the one we used during the test, so click on it.
13. On the left, Watson opens the *#local_recommend* intent with all its user examples and it highlights our example.



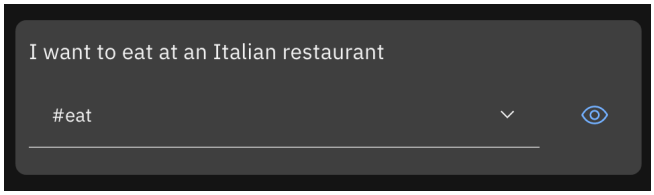
14. Select the example by checking the box next to it, then click on **Move** in order to move this example to another intent.



15. Enter *eat* as intent select it then click on **Move examples**.



Note: Now, your user example is moved to the intent *#eat*. If you will do a new test, the system will return the right intent. (Make sure that Watson finished training).



You have seen two options to improve the Watson classification:

- add a new user example in an existing intent,
- move an existing user example to the right intent.

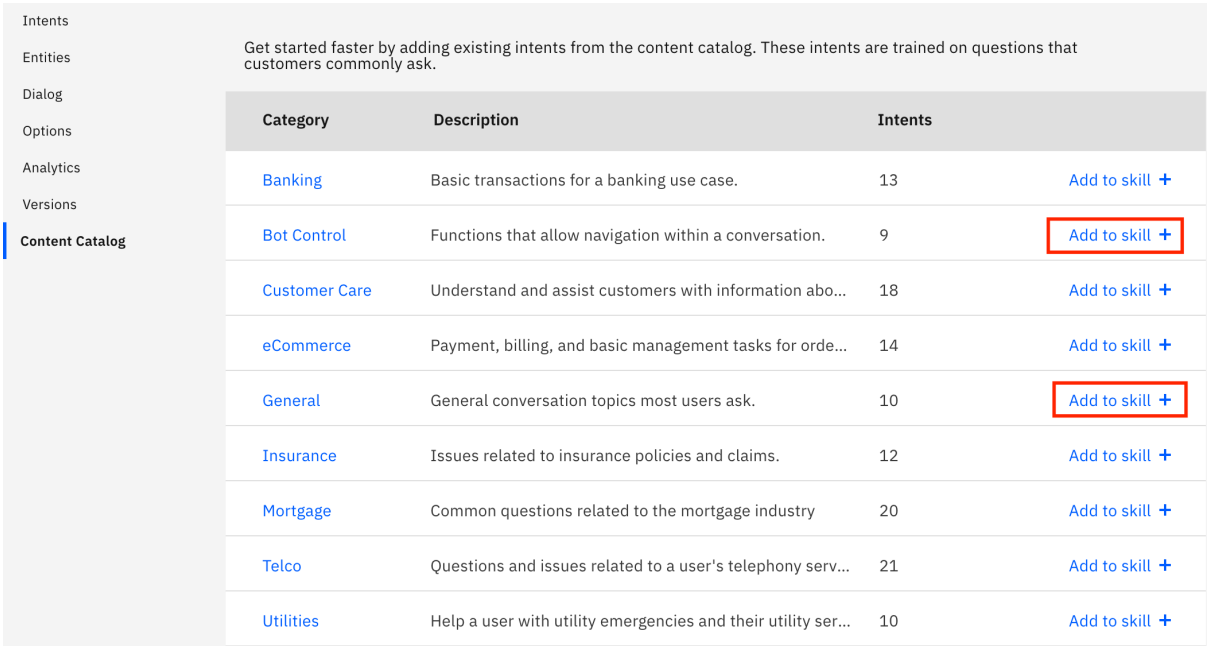
You are going to review several other options during this workshop.

13. Content Catalog

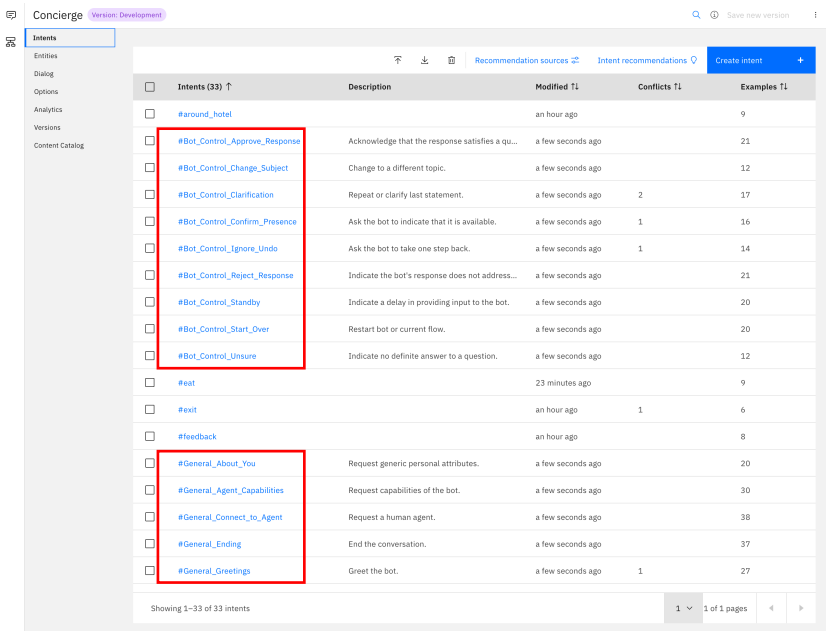
Content Catalogs provide an easy way to add common intents to your Watson Assistant service skill.

You are going to add to your Skill general conversation topics intents from the Content Catalog.

1. If you are not on your Skill page, press the left arrow on top left, to get there.
2. Click on the **Content Catalog** tab, then click **Add to skill** to add *General* and *Bot Control* categories into your training set as shown in the picture.



3. Go back to **Intents** tab, you will find new intents with prefixes *#General_* and *#Bot_Control_*. If you do not see the *#General_* intents, scroll down.



Let’s remove all useless pre-created intents.

4. Select the following intents:

   Bot_control_Change_Subject,

   Bot_control_Confirm_Presence,

   Bot_control_Ignore_Undo,

   Bot_control_Standby,

   Bot_control_Unsure,

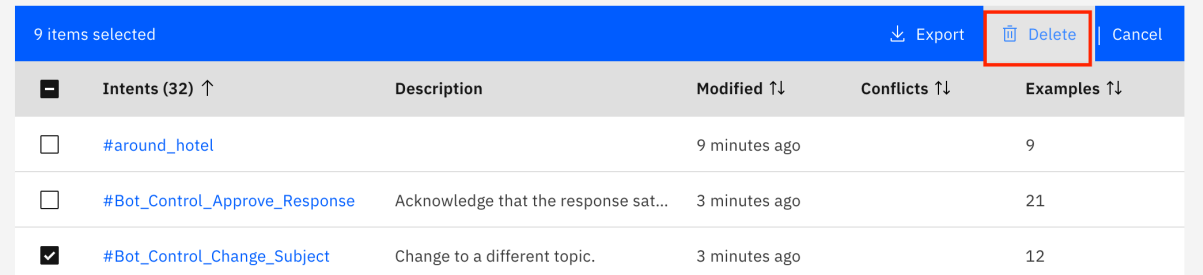
   General_Jokes,

   General_About_you,

   General_Human_or_Bot,

   General_Security_Assurance.

5. Click Delete



You should have 23 intents.

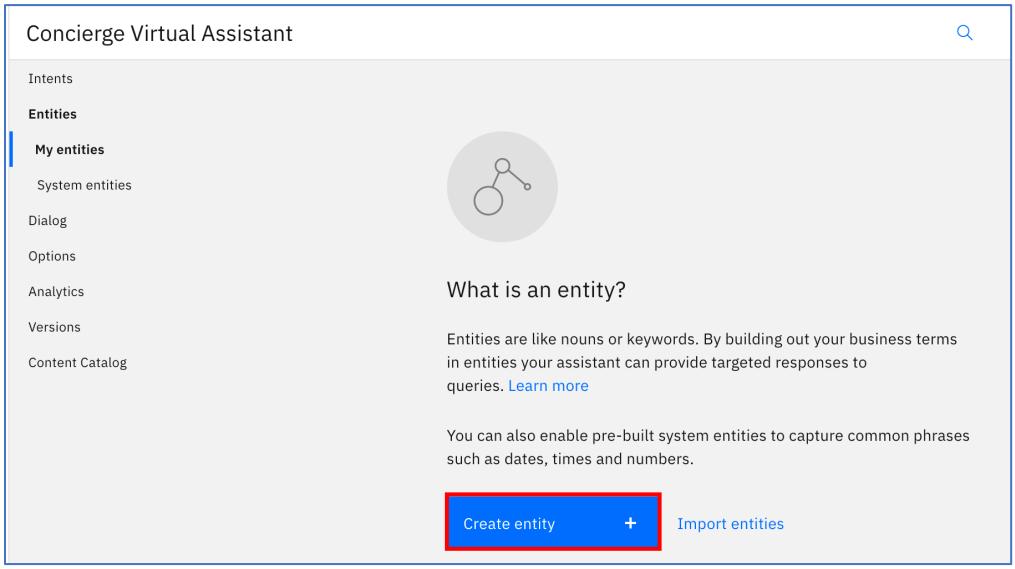
Entities

14. Create your first entity

To use the new conversation, you will need to train it with not only with intents, but also with entities and dialog nodes relevant to your application's use case. An *entity* is the portion of the user's input that you can use to provide a different response or action to an intent.

Note: for every skill, you will need to create some entities. Typically, an entity is an object or subject of our conversation. Entities can be used to help clarify users' questions/phrases. You should only create entities for things that matter and might alter the way a bot responds to an intent.

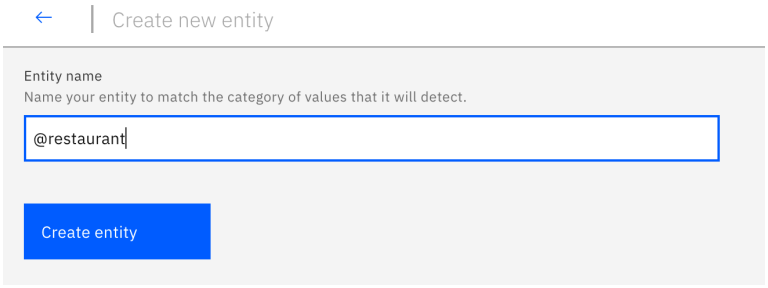
1. Click on **Entities / My entities** menu on the left-hand side.



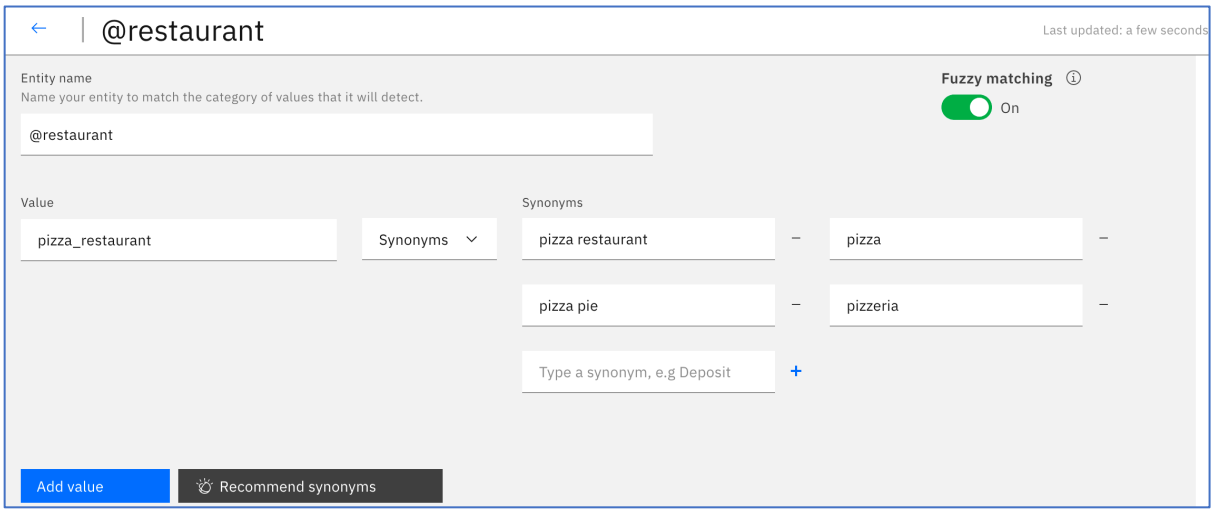
2. As you did for intent, create your first entity manually by clicking **Create entity**.

In this lab, several intents will indicate that the user is looking for a restaurant. So, you will need to create a new entity representing different restaurant categories. You will name your first entity *restaurant*, give it a value *pizza_restaurant* and enter synonyms as shown below (*pizza restaurant, pizza, pizza pie* and *pizzeria*).

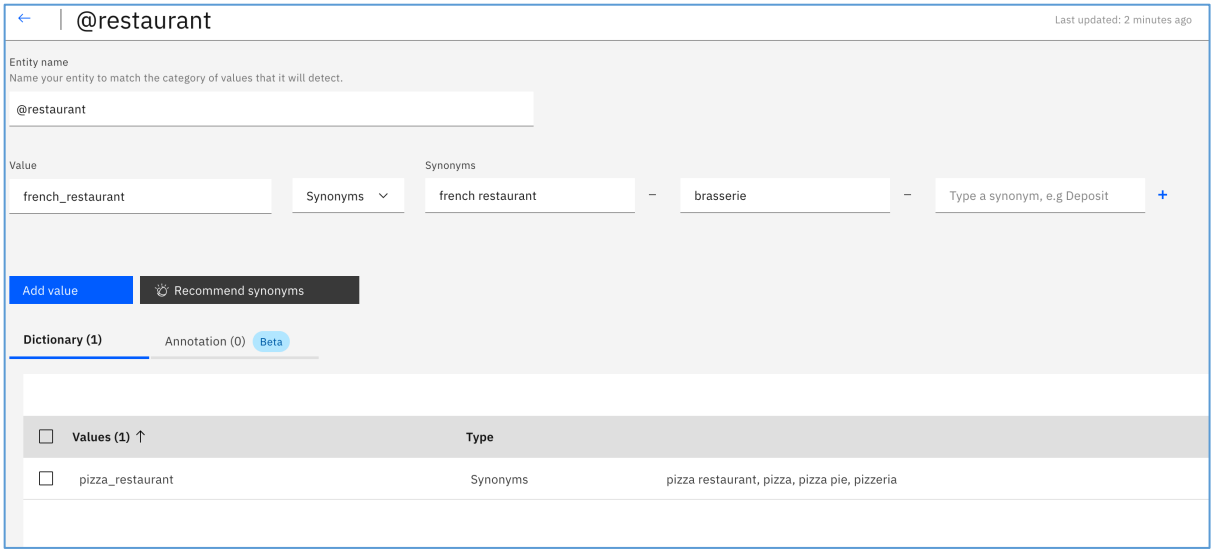
3. Type in *restaurant* as entity name, then click **Create entity**



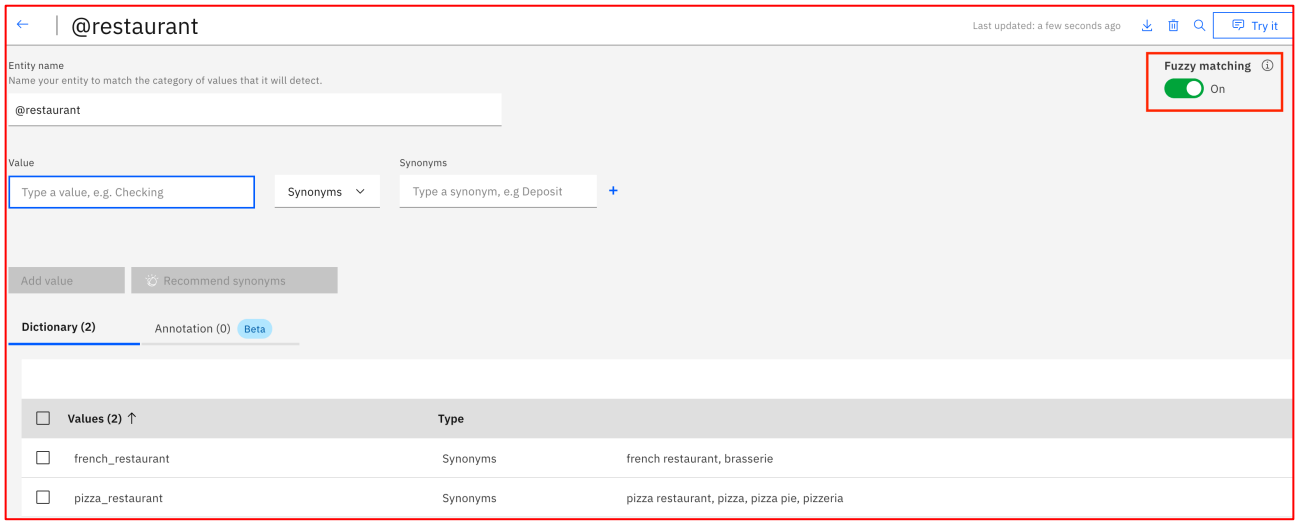
4. Fill it like below, value *pizza_restaurant* and enter synonyms *pizza restaurant, pizza, pizza pie* and *pizzeria* by clicking on plus sign next to a synonym.



5. Click **Add Value** (bottom left)
6. Fill the second value like below, value *french_restaurant* and enter synonyms *french restaurant, brasserie* by clicking on plus sign next to a synonym.



7. Click **Add Value** (bottom left)



8. Enable the **Fuzzy Matching** option if it is not already enabled. You will find it on top right (see red rectangle above).
9. When finished, go back to the Entities page by clicking on the left arrow on top.

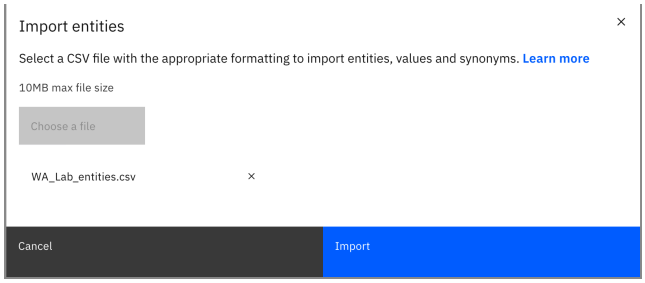
15. Import entities

Entities, like intents can be imported into the Assistant tool using a Comma Separated Values (CSV) file saved with UTF-8 encoding. This way you can quickly get many entities into your skill, rather than manually creating them.
In order to import the CSV file successfully, the file must include a minimum of two columns:

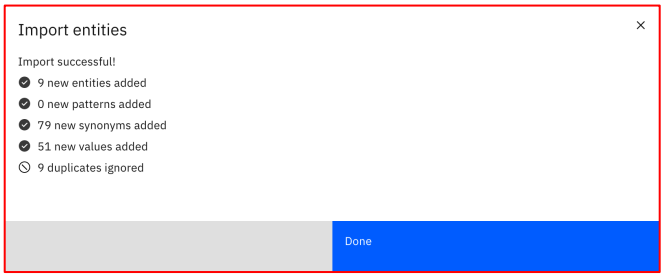
- first column for entity name
- second column for entity value
- optional columns for the synonyms. Each synonym must be in separate column

1. Return to your skill, on the Entities tab, click **Import entities** icon. (top right)

2. Click **Choose a file** then select file *WA_Lab_Entities.csv*. You should have downloaded this file from the workshop materials, it is in the folder Lab Sources.



3. Click **Upload.**

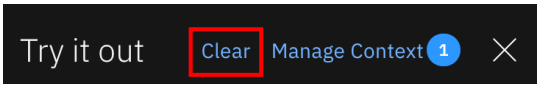


4. Click **Done.**

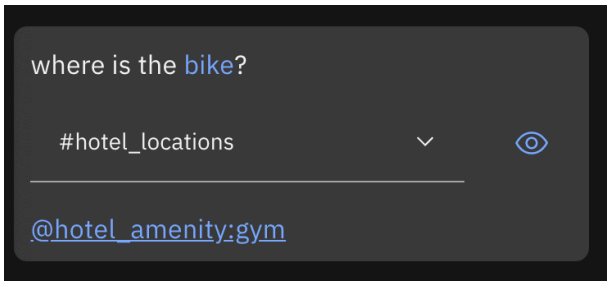
Note: the Assistant tooling did not import any examples that were exact duplicates of entities that you manually entered. So, the existing restaurant entity name, values, synonyms were ignored.

16. Test your entities

1. Open the **Try it out** panel, once Watson is done training.
2. Click on ***Clear*** to clear your panel from previous messages.

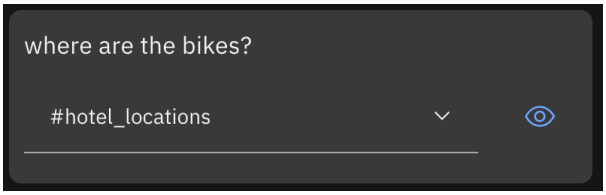


3. Type in *where is the bike?*



Note: Notice that Watson has identified the term *bike* with the correct intent, entity, and entity value (*@entity_name:entity_value)*.

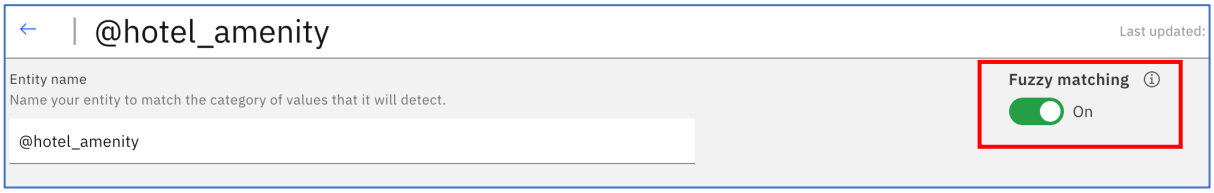
4. Type *where are the bikes?*



Note: Notice that Watson does not identify *bikes* the same as it identified *bike*. This is because entity matching is brute-force rules-based string matching without fuzzy matching.

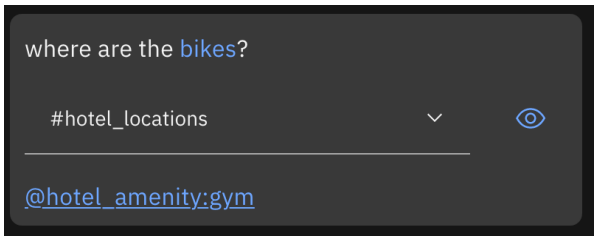
Let's turn on fuzzy matching to see how it improves entity recognition.

5. Open the *Hotel_amenity* entity and turn on **Fuzzy Matching**.



6. To complete the training, enable the **Fuzzy Matching** for the entities *@pizza-toppings* and *@pizza_notoppings*.

7. When Watson is done training, type *where are the bikes?*

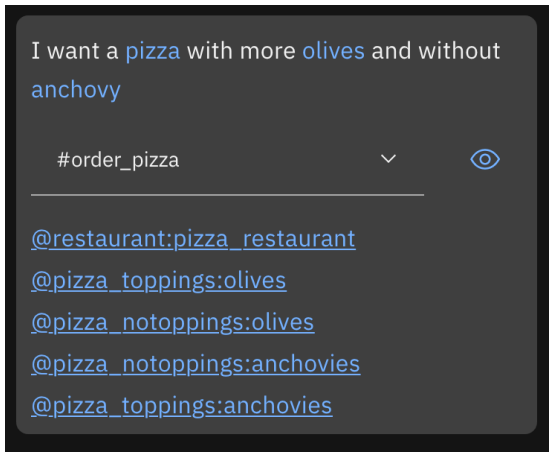


Note: notice that the correct *@entity_name:entity_value (@hotel_amenity:gym) value* pair is now recognized for *bikes*.

17. Contextual entities

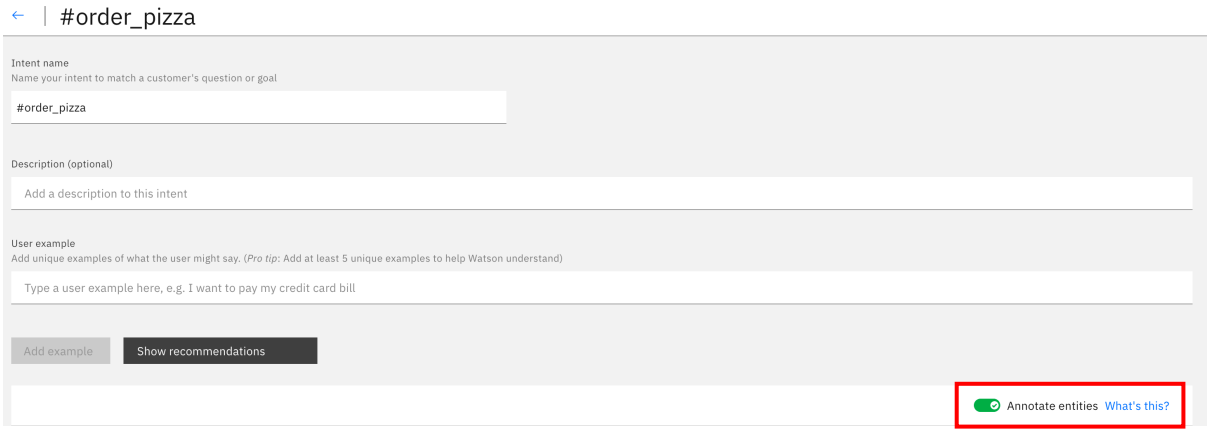
When you define specific values for an entity, the service finds entity mentions only when a term in the user input exactly matches (or closely matches if fuzzy matching is enabled) a value or synonym defined. When you define a contextual entity, a model is trained on **both the entity *value* and the *context*** in which the entity is used in sentences that you annotate. This contextual entity model enables WA to calculate a confidence score that identifies how likely a word or phrase is to be an instance of an entity, based on how it is used in the user input

1. Open **Try it out** panel.
2. Enter *I want a pizza with more olives and without anchovy* at the bottom of the chat window.

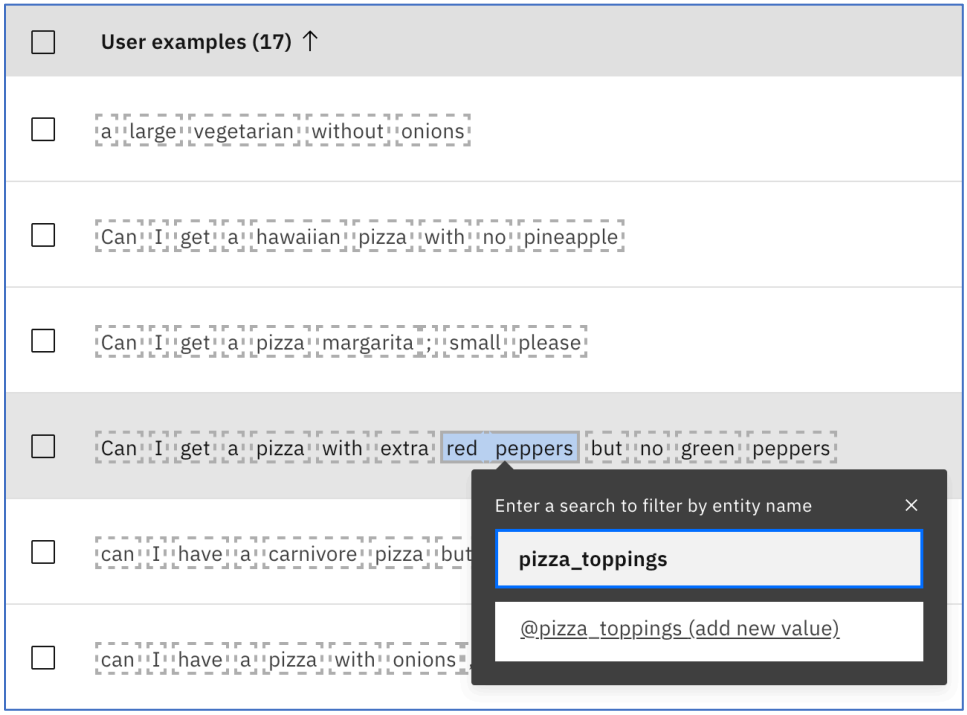


The bot understands the #order_pizza intend and the entities @pizza_toppings and @pizza_notoppings but it can't determine what the user want and don't want. This is an example where Watson needs to know the context. We are going to **annotate** the user's examples to identify the context, i.e. define how the entities are used in this particular case.

3. Go to **Intents** tab and select *#Order_pizza*
4. Enable the option **Annotate entities**



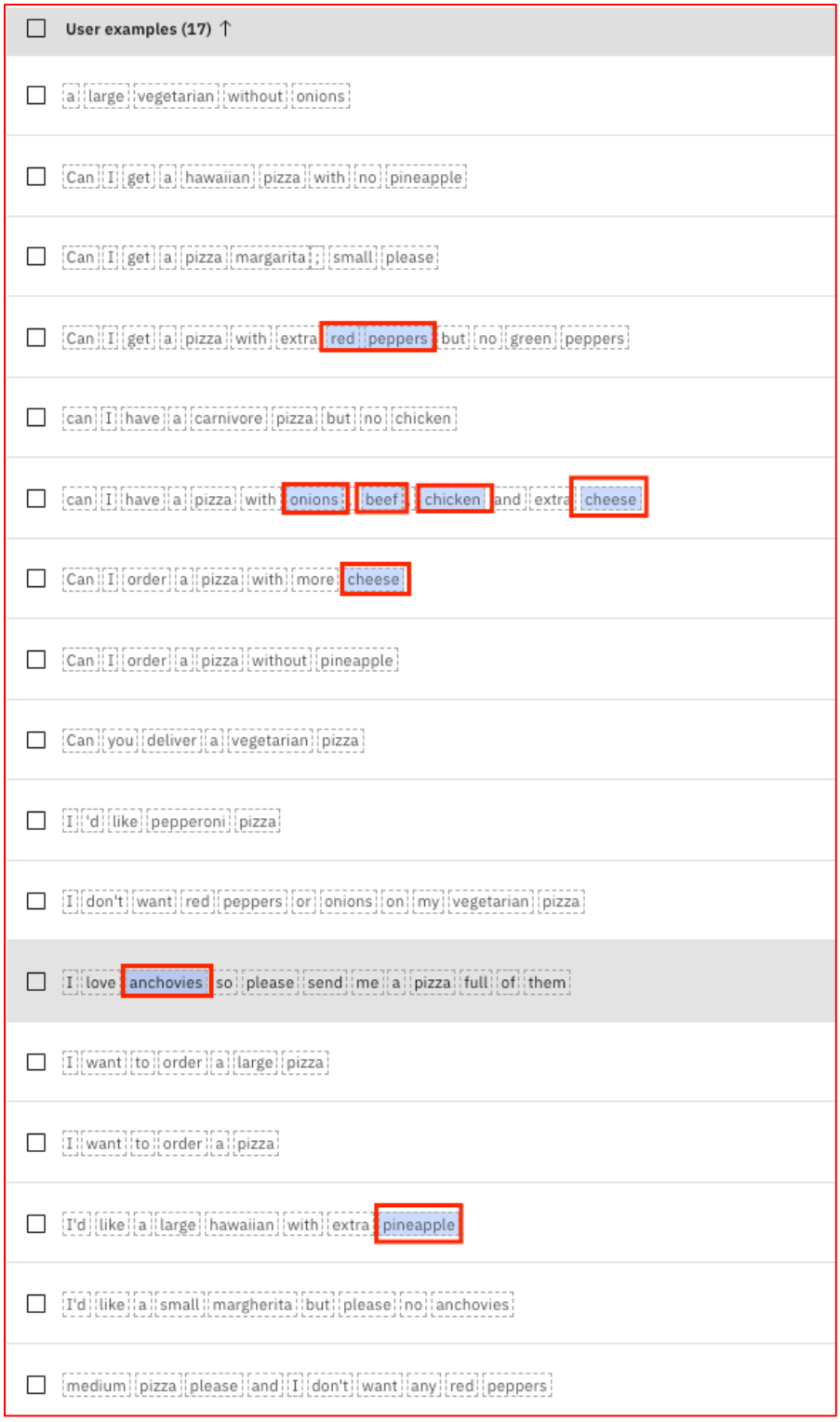
5. In the 4th example, highlight the group *red peppers* then enter *@pizza_toppings*



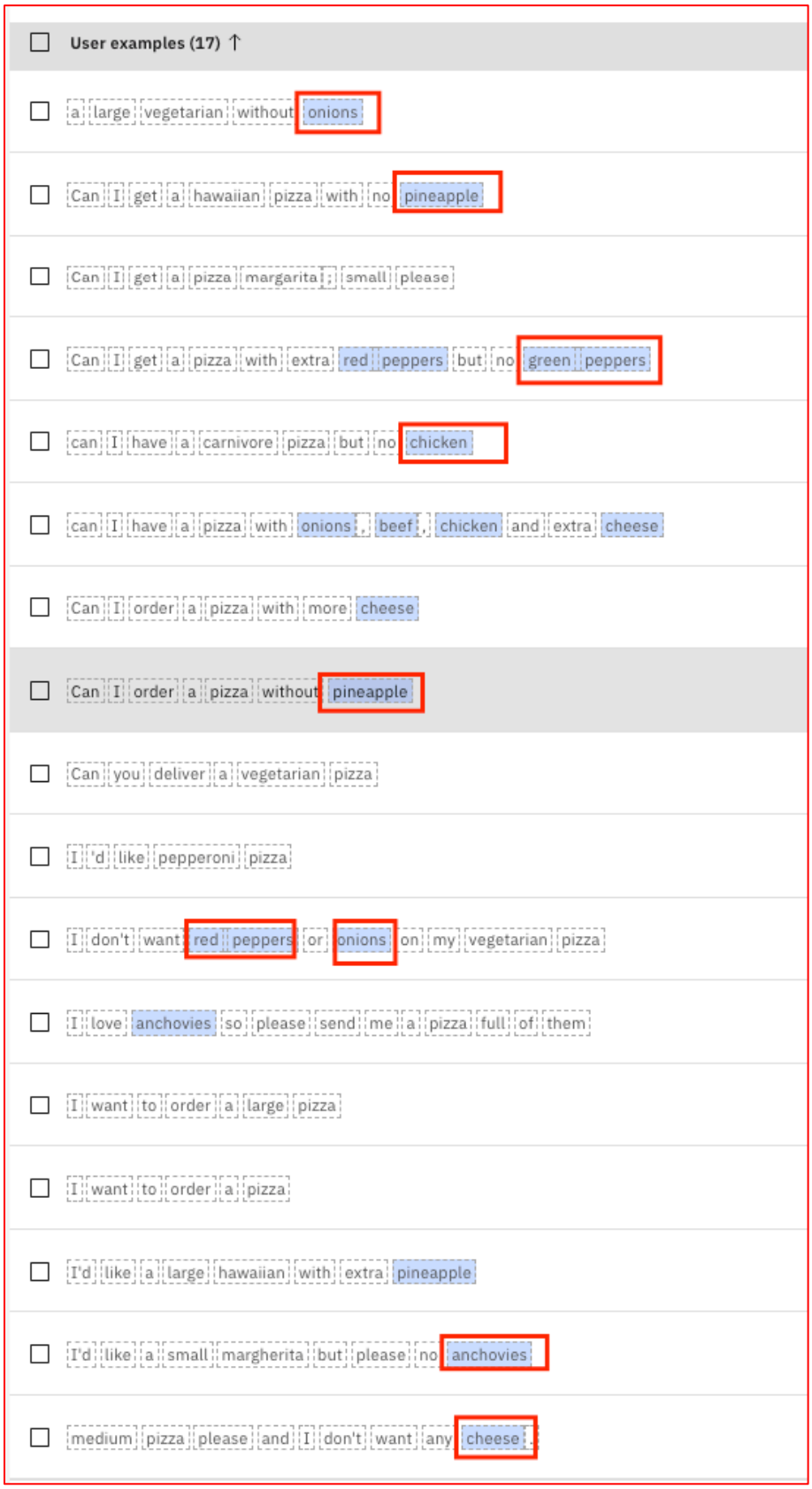
(Save your time by starting with typing, then select the right entity from the list. The list will change as you proceed typing

This way, we identified (i.e. annotated) the topping requested by the user – red peppers in this case

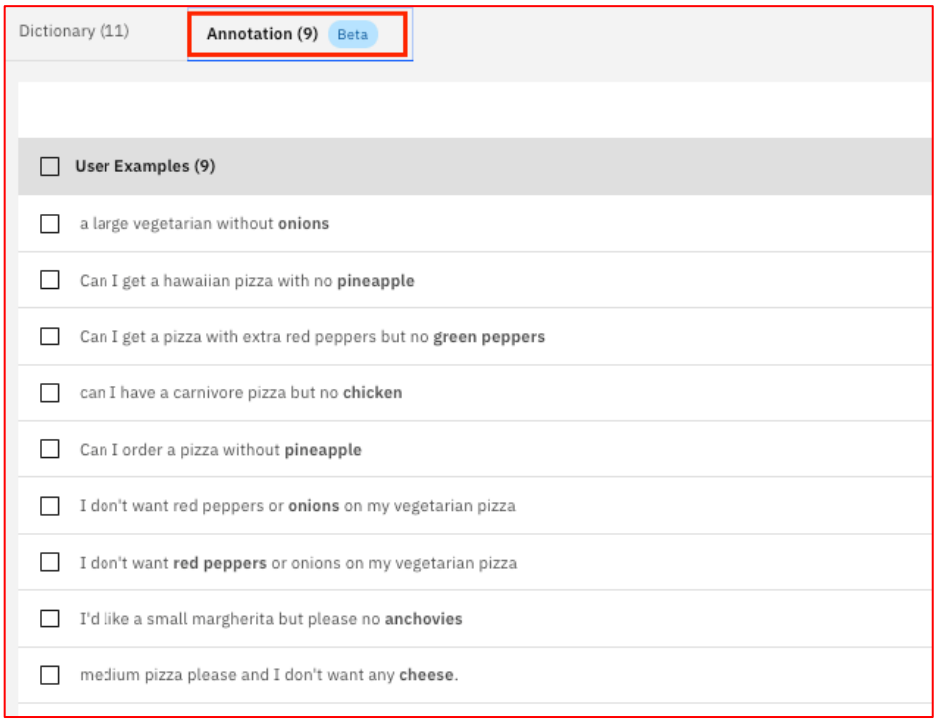
7. In a similar way, annotate these entities with *@pizza_toppings*.



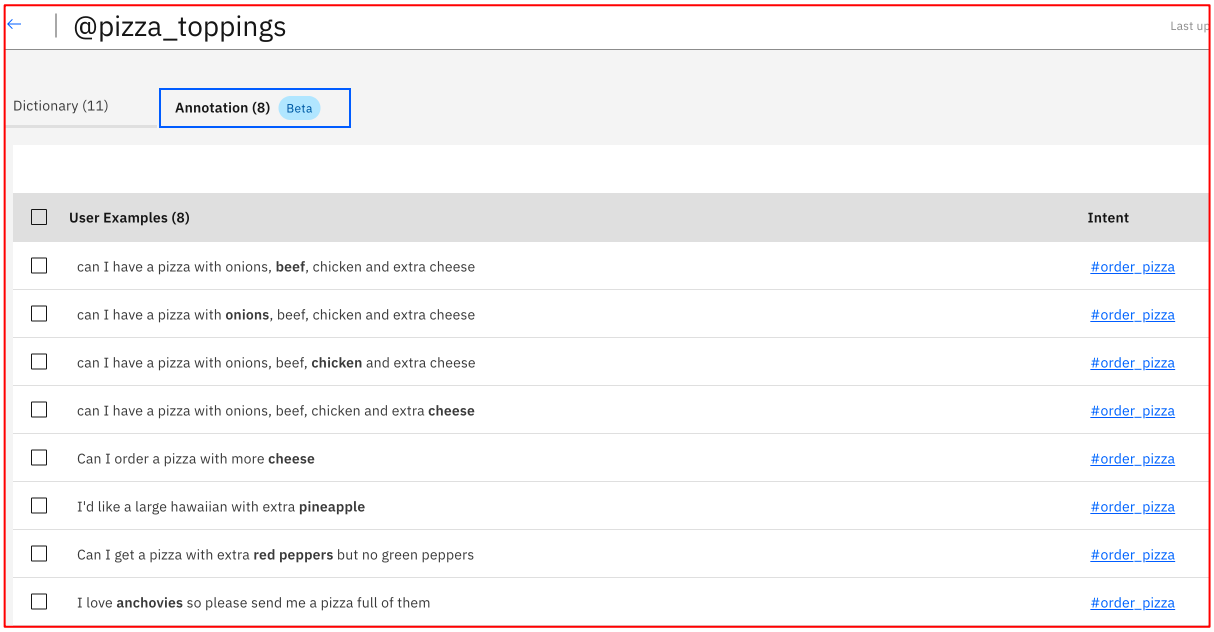
8. Now, we are going to identify the toppings that the user does ***not*** want. Annotate the intent using @pizza_notoppings entity like below:



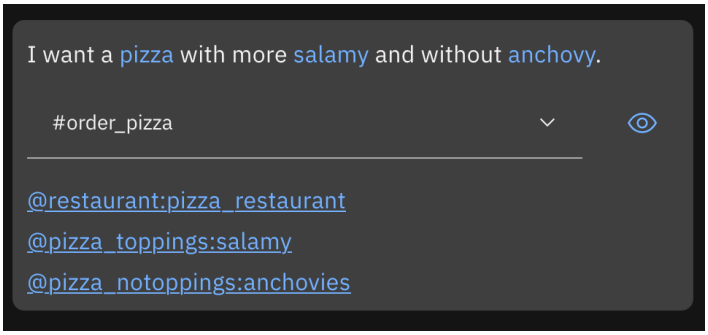
If we open the entity *@pizza_notoppings*, and the **Annotation** tab, we can see the annotated examples (9 examples)



If we open the entity *@pizza_toppings*, and the **Annotation** tab, we can see the annotated examples (8 examples)



9. Open **Try it out** panel.
10. Enter *I want a pizza with more salamy and without anchovy.*



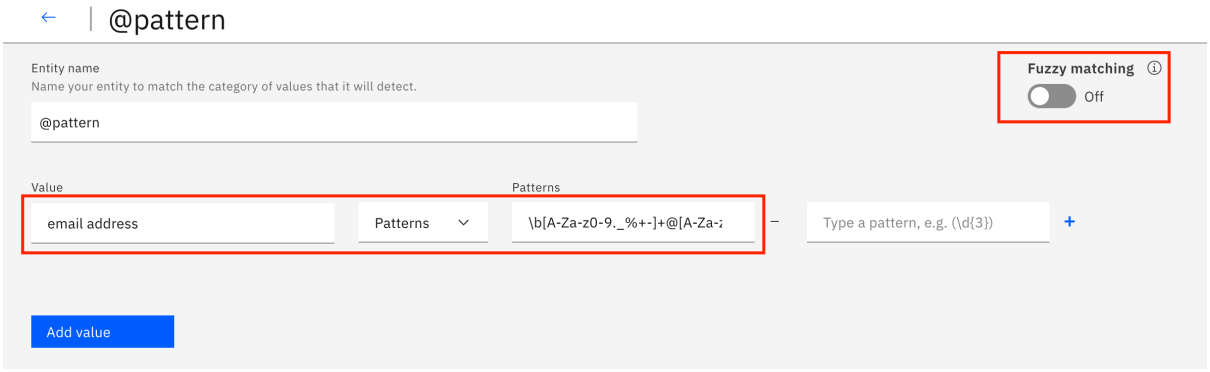
Now, Watson recognizes what user does and does not want. So, pretty cool

18. Entity pattern

In this section we will explore entity patterns.

1. Go back to the **Entities** tab, Click **Create entity**.
2. Enter *@pattern* as entity name
3. Click **Create entity**
4. Enter
    a. as value: *email address*
    b. as type select Patterns
    c. as pattern: *\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b*

    The above string is a regular expression for email address.
5. Disable the Fuzzy matching as we want WA to return the exact text defined by the regular expression.

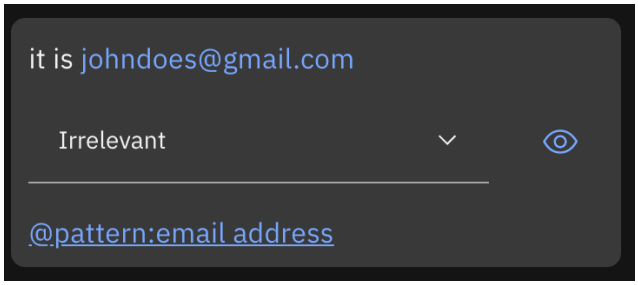


6. Click **Add value**, then click on top left arrow icon

    That's the way to extract an e-mail address from the user's input

19. Test your pattern

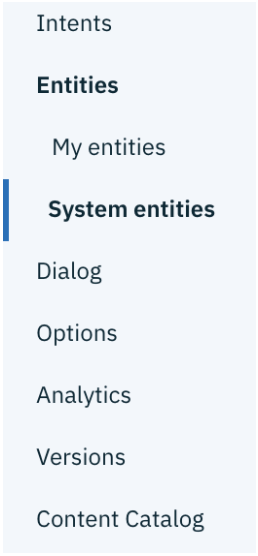
1. Open the **Try it out** panel, wait until Watson is done training.
2. Type in: *it is johndoes@gmail.com*



Watson recognizes *johndoes@gmail.com* as an entity email address.

20. Enable system entities

1. You will need to enable system entities. System entities can be used to help clarify a user's questions/phrases. If not already there, click the **Entities** and **System entities** tabs at the top of your skill.

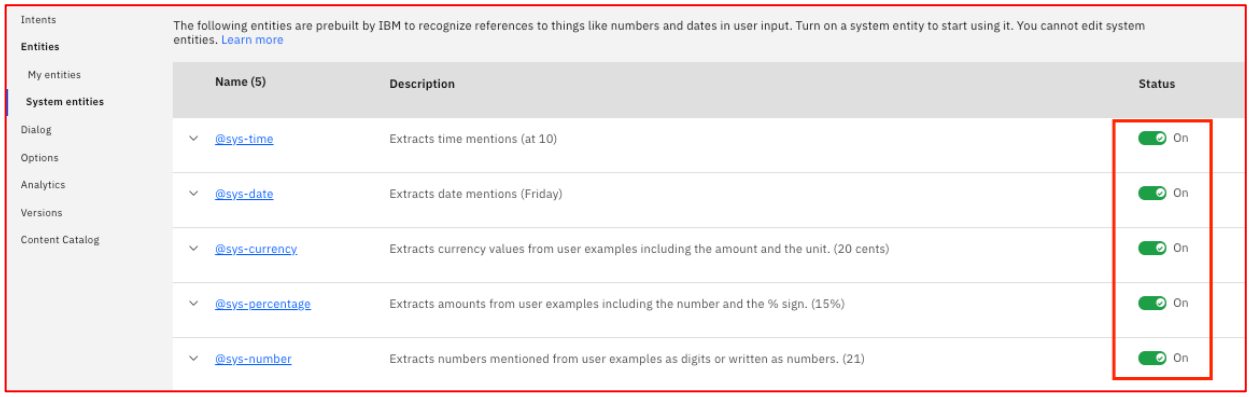


You will get these system entities

- **@sys-number:** detects numbers that are written using either numerals or words. In either case, a numeric value is returned.
- **@sys-currency:** detects monetary currency values that are expressed in an utterance with a currency symbol or currency-specific terms. A numeric value is returned.
- **@sys-percentage:** detects percentages that are expressed in an utterance with the percent symbol or written out using the word percent. In either case, a numeric value is returned.

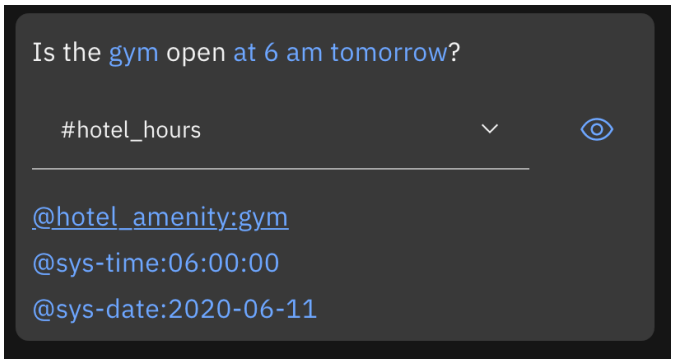
- **@sys-date, @sys-time and $timezone:** Mentions of a date range such as the weekend, next week, or from Monday to Friday are extracted as a pair of @sys-date entity mentions that show the start and end of the range

2. Switch on all system entities.



21. Test System Entities

1. Open the **try it out** panel, wait until Watson is done training.
2. Type in *Is the gym open at 6 am tomorrow?*



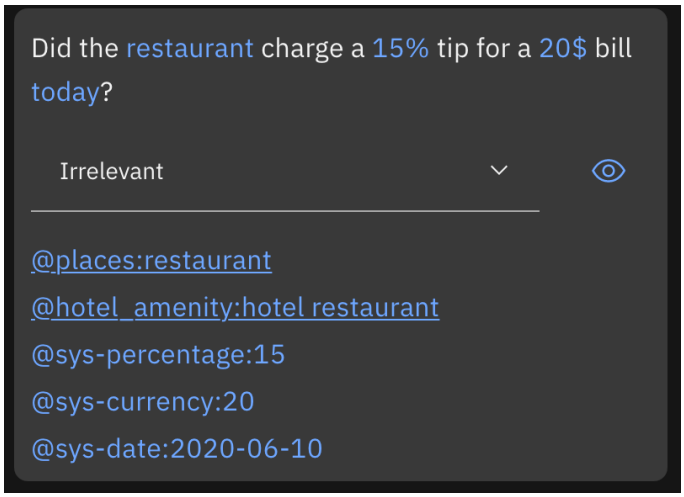
Note: notice that the time and date are recognized

6 am has been captured as **06:00:00,** the standard time format

tomorrow has been captured as **2020-06-11** (yyyy-mm-dd format)

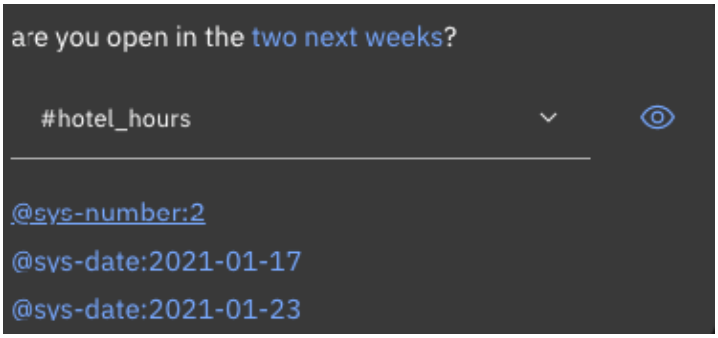
the current date is June the 10th, 2020.

3. Type in *Did the restaurant charge a 15% tip for a 20$ bill today?*



Note:The percentage, currency, number and date are recognized. You can use these system entities in your dialog node calculations the same way you would use your custom defined entities.

4. Type in *are you open the two next weeks?*



Note: Assuming today is Friday the 15th Jan, WA returns a date range ie, the first date of the period (the next Sunday), the 17th Jan and the last day of the period (the Saturday 2 weeks after), the 23rd Jan.

One option to dissociate the 2 dates is :

```
From Date: <? entities['sys-date'].filter("d", "d.role.type == 'date_from'")[0]?.value ?>
To Date: <? entities['sys-date'].filter("d", "d.role.type == 'date_to'")[0]?.value ?>
```

Conversation Dialog

22. Create your first Dialog

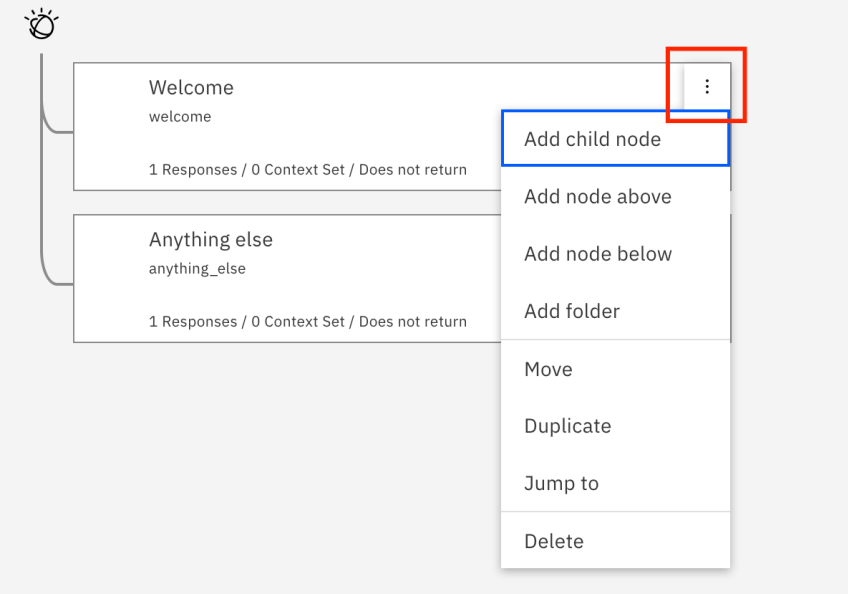
To use the new conversation, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. A dialog uses the intent and entity that have been identified, plus context from the application, to interact with the user and provide a response.

A dialog chat flow is made up of nodes. **Nodes** define the steps in the conversation or chat flow. Dialog nodes are chained together in a tree structure, and each node can be defined by two parts: a condition and a response

A **condition** is the portion of the dialog node that determines whether the node is used in the conversation. Conditions can be defined using intents, entities, context variables, and special conditions. The Spring Expression (SpEL) language is used to write valid expressions for conditions.

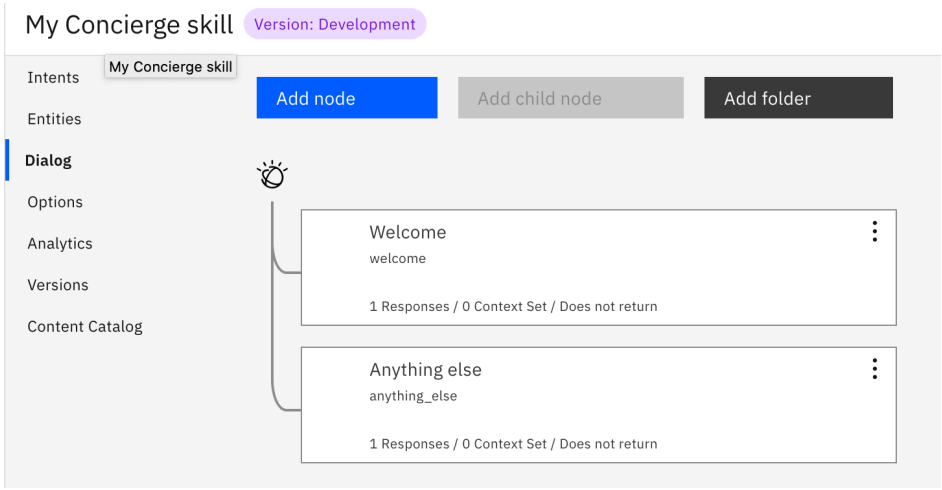
A **response** is activated if the node's condition matches the user input. A response is returned to the end-user and can be either simple text or a more advanced process (such as walking the user through the process of changing their password).

Each node has a node menu which looks like this:



This guide will go through most of the actions that can be completed from the node menu but be sure to note that **delete** will remove the node and all of its child nodes.

1. Click on **Dialog** tab

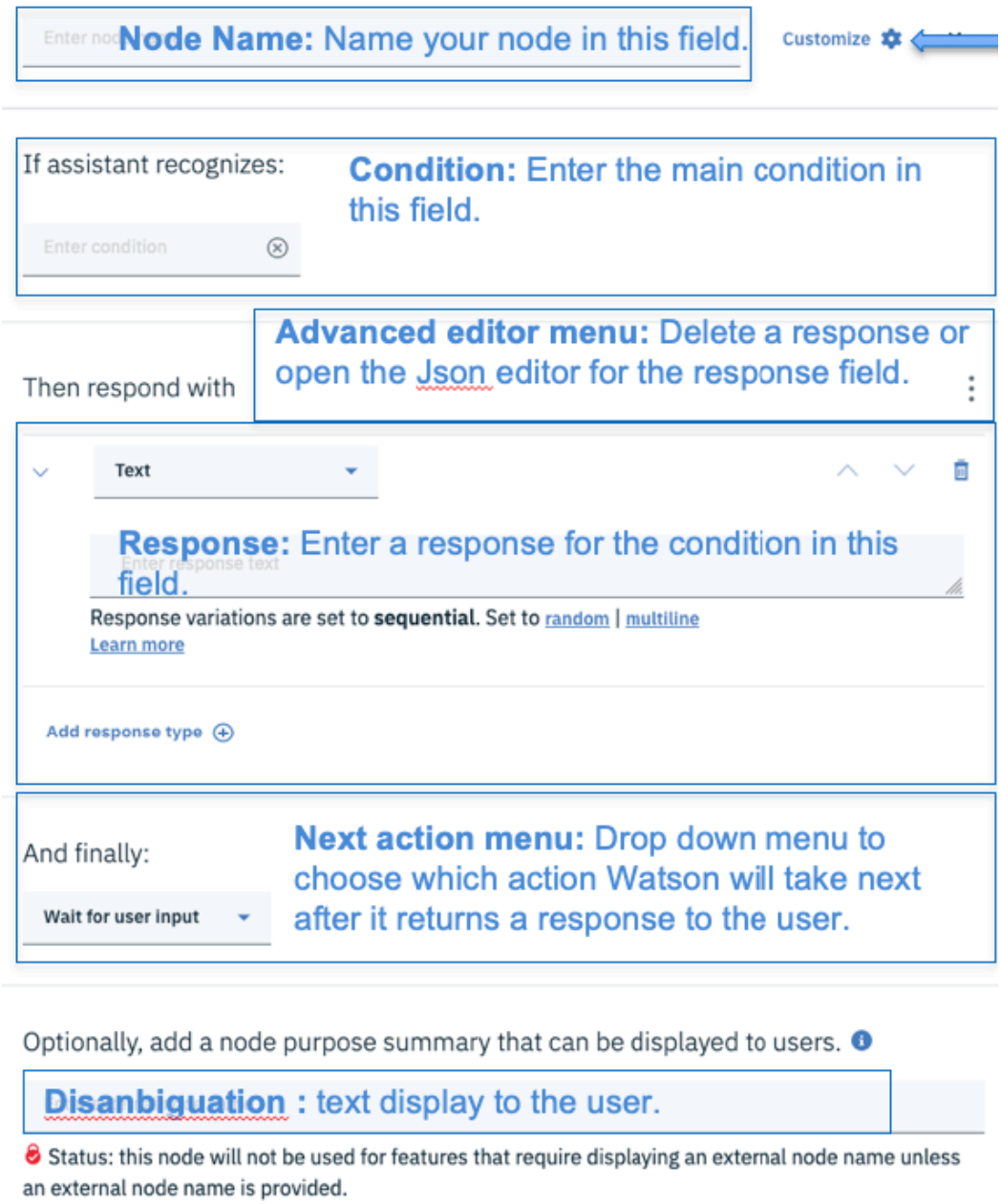


You can notice that Watson has already created two dialog nodes for you: the "**Welcome**" and the "**Anything else**" nodes.

Note:

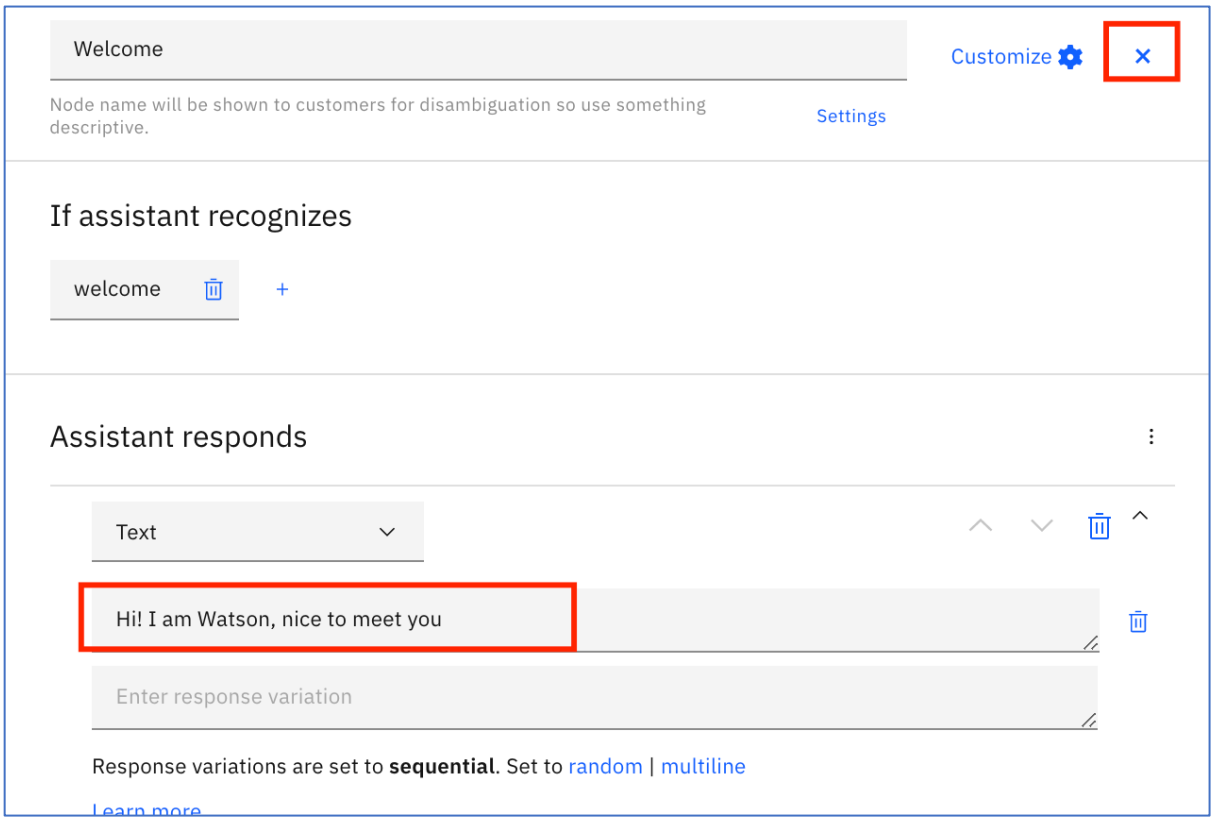
- The **Welcome** node is automatically added to your dialog chat flow. Welcome is a special condition in conversation that only evaluated as true during the first dialog turn when the application does not contain any user input. If the initial request from the application contains user input, Welcome node will not be triggered. Typically, welcome is used when you always want to display a message or a greeting at the beginning of the conversation.
- The **Anything else** node is also automatically added to you dialog chat flow. The Anything else node is set to always evaluate to true. Its purpose is to always provide a response to the user if the user's input does not evaluate to true for any other parent node in the tree.

- The panel that appears on the **right side of the screen** is the editor panel for the node. When you create a new node or when you click on a node to make edits, this panel will appear for the node that you click on. This is where all editing is done for the node. The image below explains the elements of the node editor.



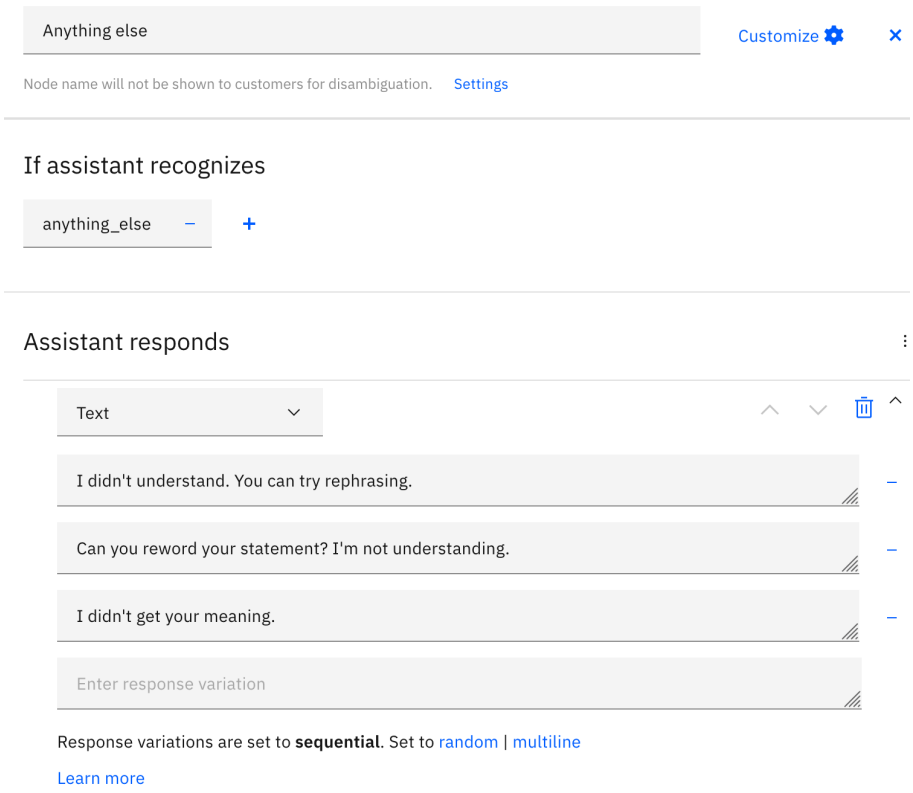
2. Click on the **Welcome** node to edit it and fill it as below

Field	Value
Node name	*Welcome*
If assistant recognizes	*welcome*
Assistant responds	*Hi! I am Watson, nice to meet you*



3. Close the **Welcome** node by clicking on **x** on top right.

4. Click on the **Anything else** node, you will see 3 pre-created responses.



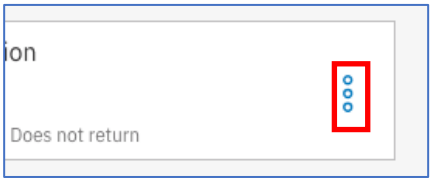
You may edit these if you wish.

5. Close the Anything else node by clicking on **x** on top right.

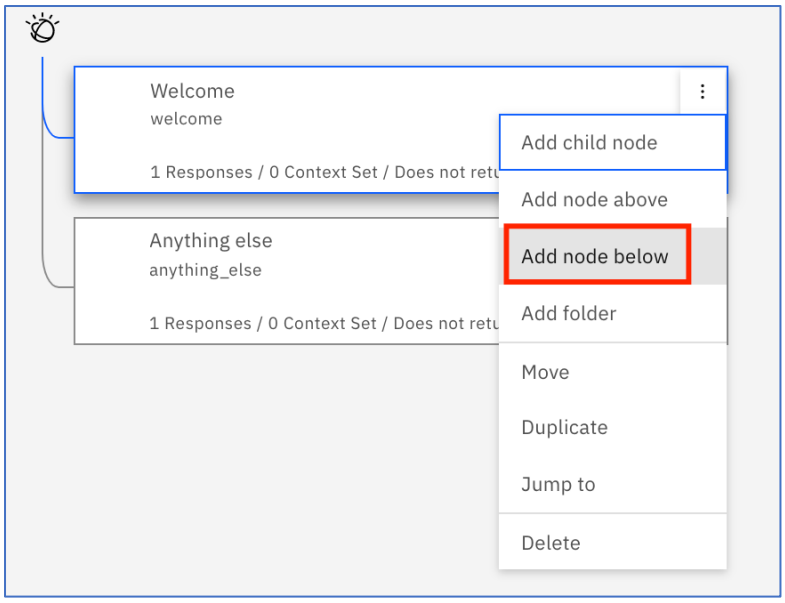
23. Create your first nodes

You can create a dialog branch for each of the *intents* you identified as well as the start & end of the conversation. You should also greet the end user as appropriate. For the greeting, you can create a new dialog node to respond to a *greeting* intent. To do this you are going to leverage the intent *#General_Greetings* that you have imported in earlier steps.

Note: When you click on **Add node** button at the top of the tab, a node will be added below the selected node. Which means you must know which node is the selected node. Thus, the best way to add a node into correct position in the dialog flow is to use a **node menu** – the 3 vertical dots on the right hand side of a node**.**



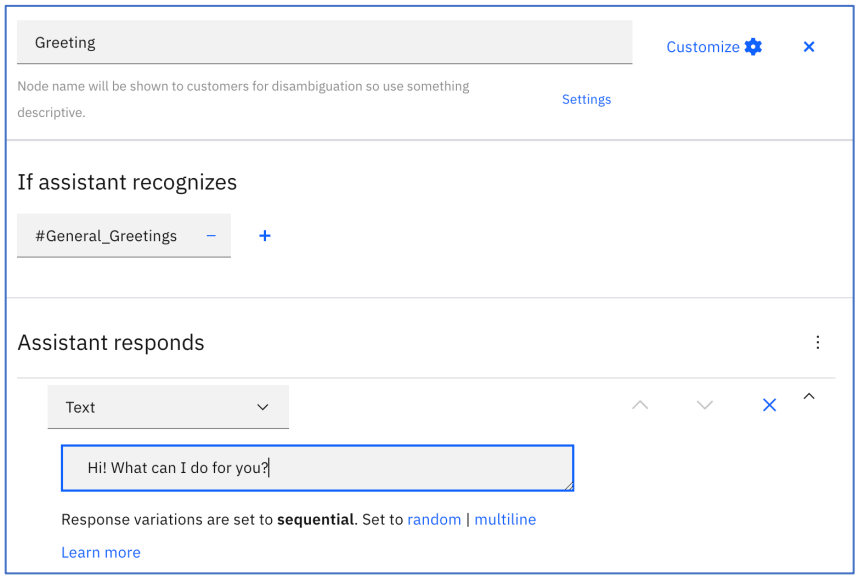
1. On the **start of the conversation** node, the **node menu** (3 dots) then click **Add node below.**



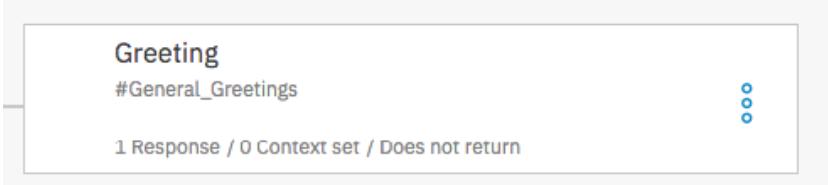
2. In this new node, enter the following values.

Field	Value
Node name	*Greeting*
If assistant recognizes	*#General_Greetings*
Assistant responds	*Hi! What can I do for you?*

By setting the condition (if assistant recognizes) to an *intent*, you are indicating that this node will be triggered by any input that matches the specified *intent*.



3. Close the node, by clicking **x** in the top right corner. Your new node should look like this:



Note: The figure above indicates that there is only one condition that is evaluated (the #General_Greeting intent), one response and no context variable used.

You will create a dialog branch to respond to the *#hotel_hours* intent. Because there are multiple possibilities to manage this intent, this branch will be more complex. For the beginning, we will keep it simple

4. Add a node below the **Greeting** node as below

Field	Value
Node name	*Hotel Hours*
If assistant recognizes	*#hotel_hours*
Assistant responds	*The @hotel_amenity is open from 6 am to 9 pm*

@hotel_amenity is the entity captured by Watson. The bot will return this information and demonstrate that it understood the request.

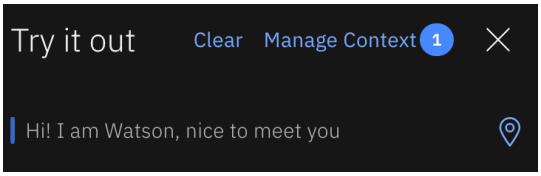
24. Test your conversation

Now, you should have a dialog with four nodes.

it's time to test it.

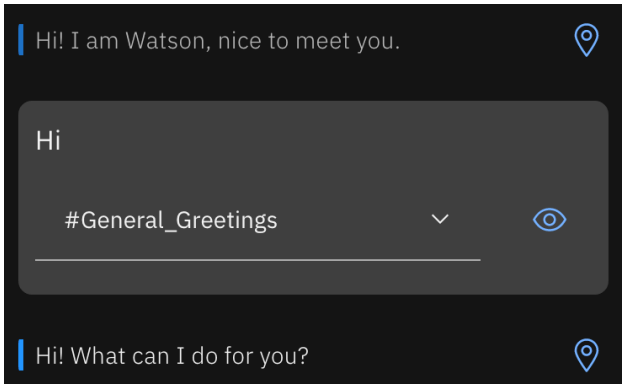
11. Open **Try it out** panel and click **Clear** on the top of the panel.

A test user interface will immediately launch and, based on the root node you just created, provide a greeting to the end user. (You may see a message that Watson is being trained.)

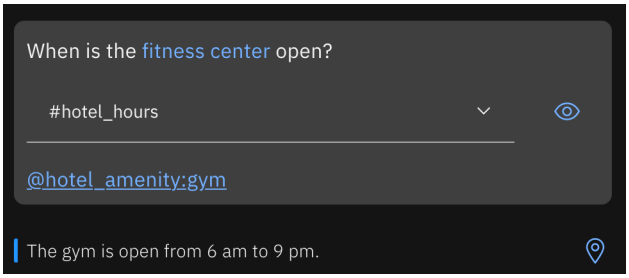


12. Enter *Hi* at the bottom of the chat window.

    The bot understands the #General_greetings intend and send you back the right response.

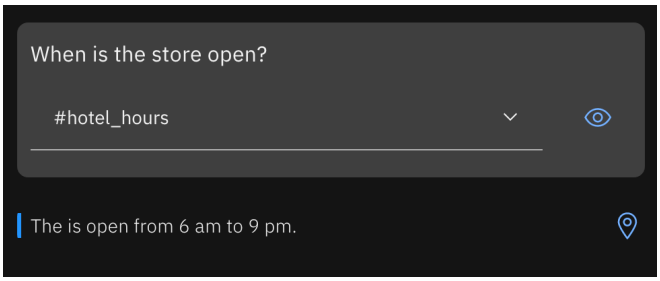


13. Enter *When is the **fitness center** open?*



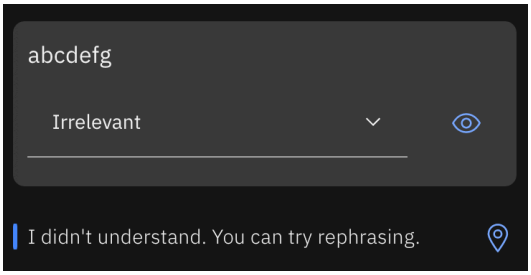
Note: Watson should return the response that is matched to *#hotel_hours* and *@hotel_amenity:gym* as *fitness center* is a synonyme of this entity.

14. Enter *When is the **store** open?*



Note: Watson understood the intent *#hotel_hours*. But as *store* is not defined as value of any entities, the system returns the incomplete answer without the *@hotel_amenity* value. During the next lab, we are going to understand how to improve the WA behaviour to manage such behaviour.

15. Enter *abcdefg*



Watson cannot identify anything so it executes the **anything_else** node.

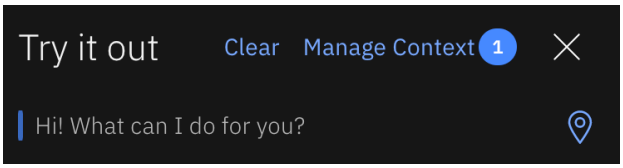
Enrich your data by using context variables

Watson Assistant allows you to store information. This is done using the **context**. For example, when asked where something is, you may wish to show a map. In such case, your dialog would store the user's location in the context, and your application can use it later to show that location on a map. You can set variables in the context using the **Json editor** or the **context editor**.

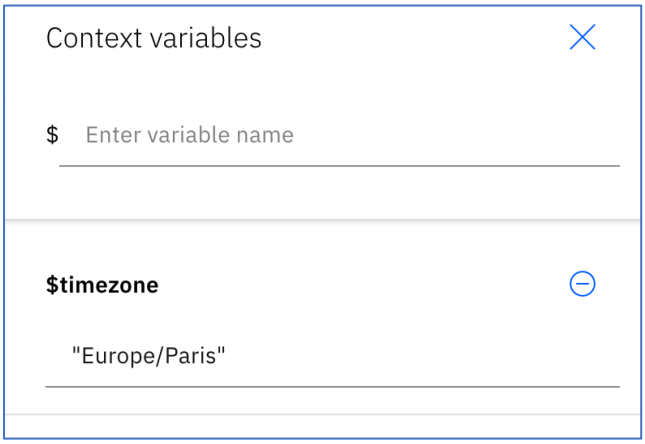
There are two methods to access a context variable:

- Shorthand: $variable_name
- Full Syntax: context.variable_name

1. To view the context variables that are currently set in a conversation, open the **Try it out** panel, and click on **Manage Context.** The number besides Manage Context tells you how many context variables are currently set.



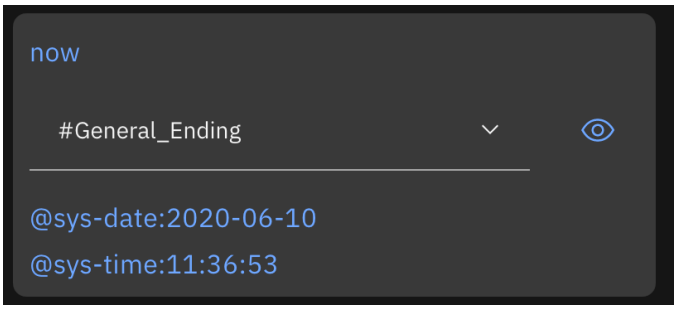
This will open the **Context variables** panel.



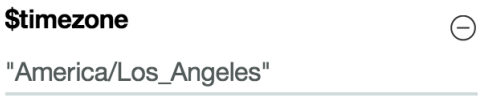
Since you enabled sys_date and sys_time system entities, Watson stores the user's time zone (which comes from the browser) into the $timezone context variable.

2. Close **Context variables** by pressing the X sign on top right.

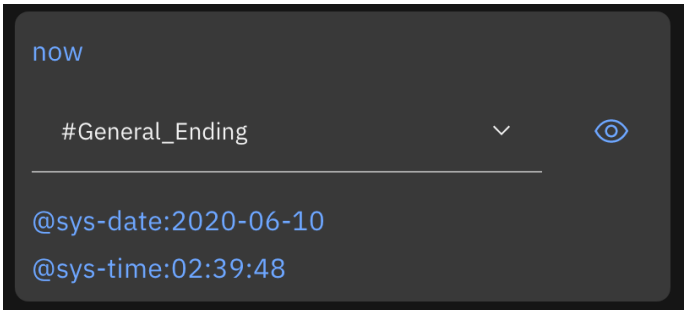
3. In the **try it out** panel, enter *now*



4. You will see the date and time in your time zone. Go back to **Manage Context** panel and change the *$timezone* variable to something else.
5. Write down (or copy into clipboard) your current time zone, so you can use it later. Replace the time zone with *America/Los_Angeles*



6. Close the Context variables panel. In the **Try it out** panel, enter again *now*



You will see that WA applied the new time zone.

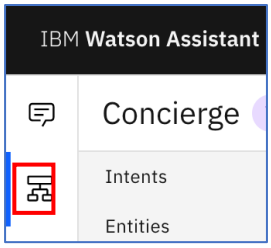
7. To avoid any confusion in the time zone, set back the time zone as it was originally.

Save your skill to your computer

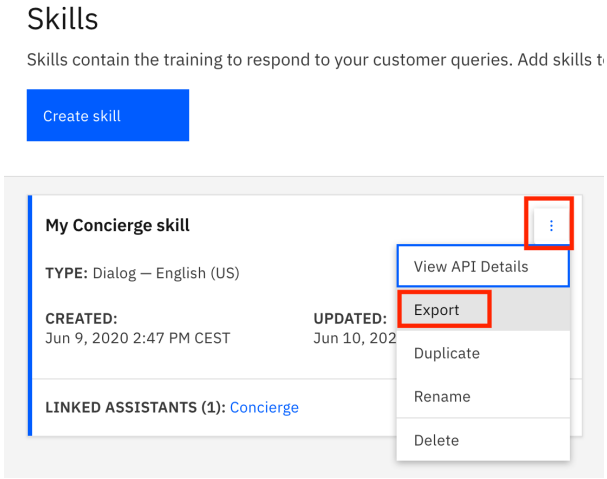
(optional)

For backup purposes, you might want to save your Skill on your local machine.

1. On top left, click on the **Skills** icon.



2. Click on the Skill's menu (3 dots), then select Export.

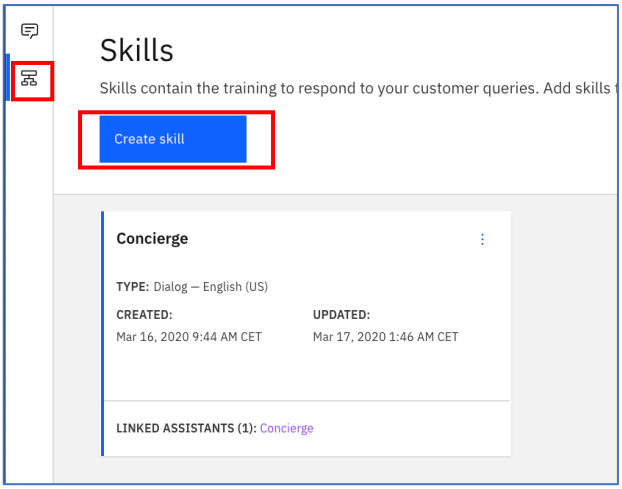


3. In the **Save As** field, keep the proposed name (skill-Concierge) and click **Save**. With this you saved your skill into a JSON file on your computer. At any time, you can retrieve this Skill back to WA by importing this JSON file.

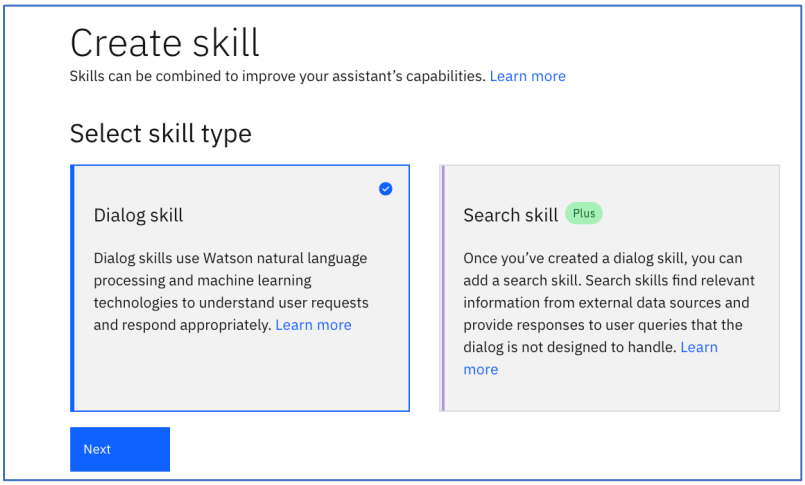
Note: if you want to compare your skill with the one we created, feel free to import into your Assistant our skill as it was after the end of this exercise. This skill is in file *skill-My-Concierge-Skill-Lab1.json*.

You can import a skill into your Assistant this way:

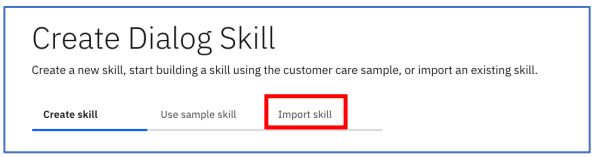
1. On Skills tab, click Create skill.



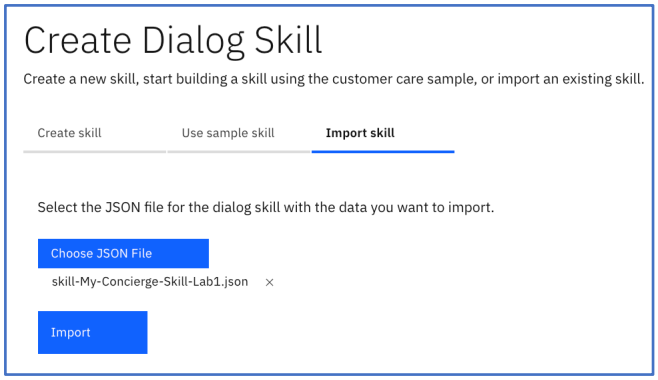
2. Select **Dialog skill** and click **Next**.



3. Click on **Import skill**.



4. Click **Choose JSON File**, then select the file with the exported skill.



5. Click **Import**.
6. Now, you can see the skill you just imported on your **Skills** page.