

Dialogflow Components

Agents

A Dialogflow agent is a virtual agent that handles conversations with your end users. It is a natural language understanding module that understands the nuances of human language. Dialogflow translates end user text or audio during a conversation to structured data that your apps and services can understand. You design and build a Dialogflow agent to handle the types of conversations required for your system.

A Dialogflow agent is similar to a human call center agent. You train them both to handle expected conversation scenarios, and your training does not need to be overly explicit.

Intents

An intent categorizes an end-user's intention for one conversation turn. For each agent, you define many intents, where your combined intents can handle a complete conversation. When an end-user writes or says something, which is referred to as an end-user expression, Dialogflow matches the end-user expression to the best intent in your agent. Matching an intent is also known as intent classification.

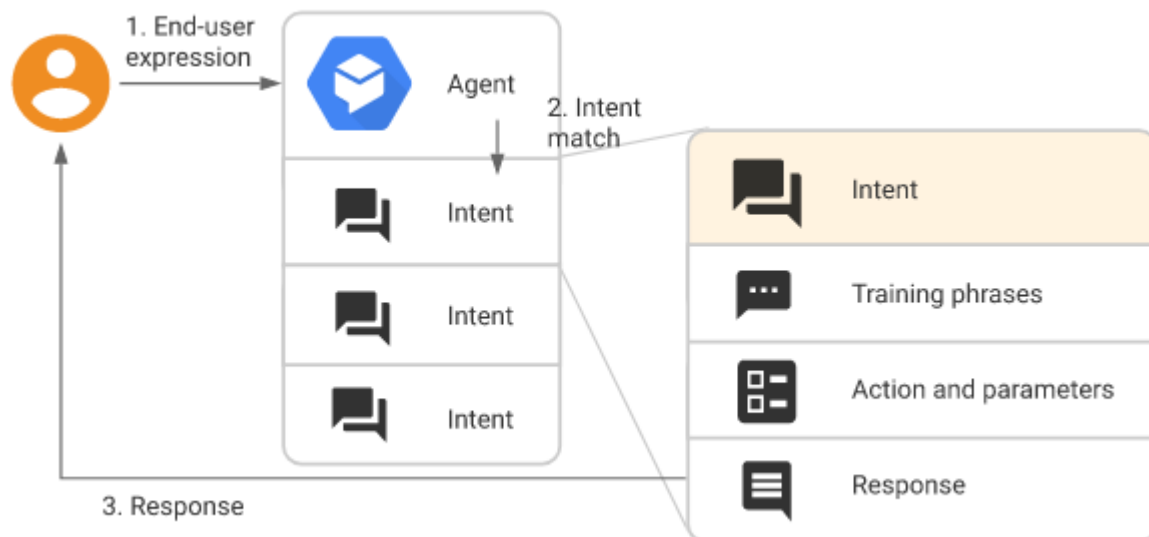
For example, you could create a weather agent that recognizes and responds to end user questions about the weather. You would likely define an intent for questions about the weather forecast. If an end user says "What's the forecast?", Dialogflow would match that end user expression to the forecast intent. You can also define your intent to extract useful information from the end user expression, like a time or location for the desired weather forecast. This extracted data is important for your system to perform a weather query for the end user.



A basic intent contains the following:

- 1.Training phrases:** These are example phrases for what end users might say. When an end user expression resembles one of these phrases, Dialogflow matches the intent. You don't have to define every possible example, because Dialogflow's built-in machine learning expands on your list with other similar phrases.
- 2.Action:** You can define an action for each intent. When an intent is matched, Dialogflow provides the action to your system, and you can use the action to trigger certain actions defined in your system.
- 3.Parameters:** When an intent is matched at runtime, Dialogflow provides the extracted values from the end user expression as parameters. Each parameter has a type, called the entity type, which dictates exactly how the data is extracted. Unlike raw end user input, parameters are structured data that can easily be used to perform some logic or generate responses.
- 4.Responses:** You define text, speech, or visual responses to return to the end user. These may provide the end user with answers, ask the end user for more information, or terminate the conversation.

The following diagram shows the basic flow for intent matching and responding to the end-user:



Entities

Each intent parameter has a type, called the entity type, which dictates exactly how data from an end user expression is extracted.

Dialogflow provides predefined system entities that can match many common types of data. For example, there are system entities for matching dates, times, colors, email addresses, and so on. You can also create your own custom entities for matching custom data.

Contexts

Dialogflow contexts are similar to natural language context. If a person says to you "they are orange", you need context in order to understand what "they" is referring to. Similarly, for Dialogflow to handle an end user expression like that, it needs to be provided with context in order to correctly match an intent.

Using contexts, you can control the flow of a conversation. You can configure contexts for an intent by setting input and output contexts, which are identified by string names. When an intent is matched, any configured output contexts for that intent become active. While any contexts are active, Dialogflow is more likely to match intents that are configured with input contexts that correspond to the currently active contexts.

Follow-up intents

You can use follow-up intents to automatically set contexts for pairs of intents. A follow-up intent is a child of its associated parent intent. When you create a follow-up intent, an output context is automatically added to the parent intent and an input context of the same name is added to the follow-up intent. A follow-up intent is only matched when the parent intent is matched in the previous conversational turn. You can also create multiple levels of nested follow-up intents.

Dialogflow provides many predefined follow-up intents for common end user replies like "yes", "no", or "cancel". You can also create your own follow-up intents to handle custom replies.

Fulfillment for integrations

By default, your agent responds to a matched intent with a static response. If you're using one of the integration options, you can provide a more dynamic response by using fulfillment. When you enable fulfillment for an intent, Dialogflow responds to that intent by calling a service that you define. For example, if an end-user wants to schedule a haircut on Friday, your service can check your database and respond to the end-user with availability information for Friday.

Each intent has a setting to enable fulfillment. If an intent requires some action by your system or a dynamic response, you should enable fulfillment for the intent. If an intent without fulfillment enabled is matched, Dialogflow uses the static response you defined for the intent.

When an intent with fulfillment enabled is matched, Dialogflow sends a request to your webhook service with information about the matched intent. Your system can perform any required actions and respond to Dialogflow with information for how to proceed. The following diagram shows the processing flow for fulfillment.

Design your intents

Take five minutes and write down the intents for the lab use case. The key here is the training phrases associated with the intents: These are example phrases for what end-users might say. In this use case they can be as follows:

Intent: Change a name on a reservation

Training Phrases:

1. I want to change name on my itinerary
2. Change name on my reservation
3. Change name
4. I want to change my name on my hotel reservation

Think of other possibilities and write them down. The key here is to increase call availability, remove complicated menu systems and achieve shorter handling times. By having Contact Center AI handle these intents we can achieve these goals.

Setup and requirements

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you

new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

To complete this lab, you need:

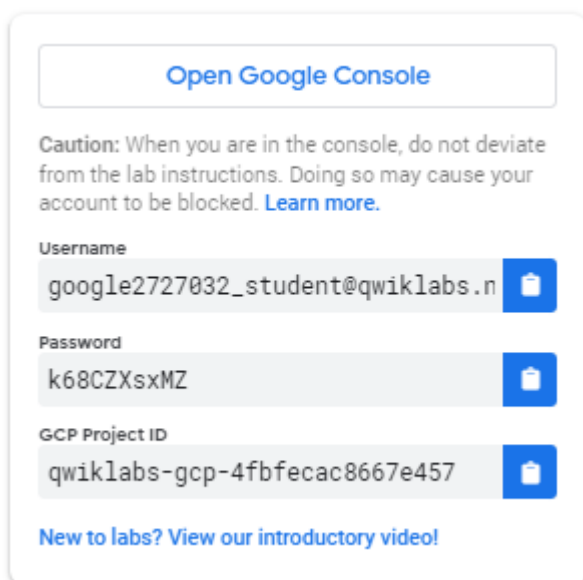
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

Note: If you are using a Chrome OS device, open an Incognito window to run this lab.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



Open Google Console

Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)

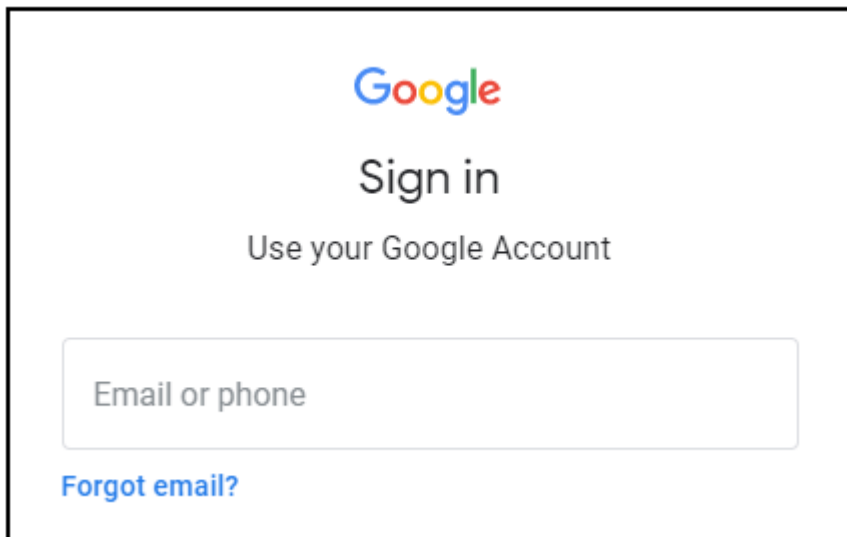
Username
google2727032_student@qwiklabs.n

Password
k68CZXsxMZ

GCP Project ID
qwiklabs-gcp-4fbfecac8667e457

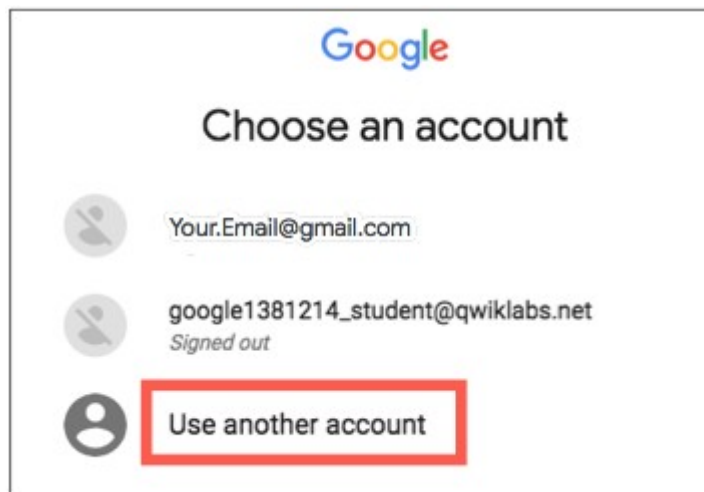
[New to labs? View our introductory video!](#)

2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Tip: Open the tabs in separate windows, side-by-side.

Choose an account page, click **Use Another**



Account.

3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:

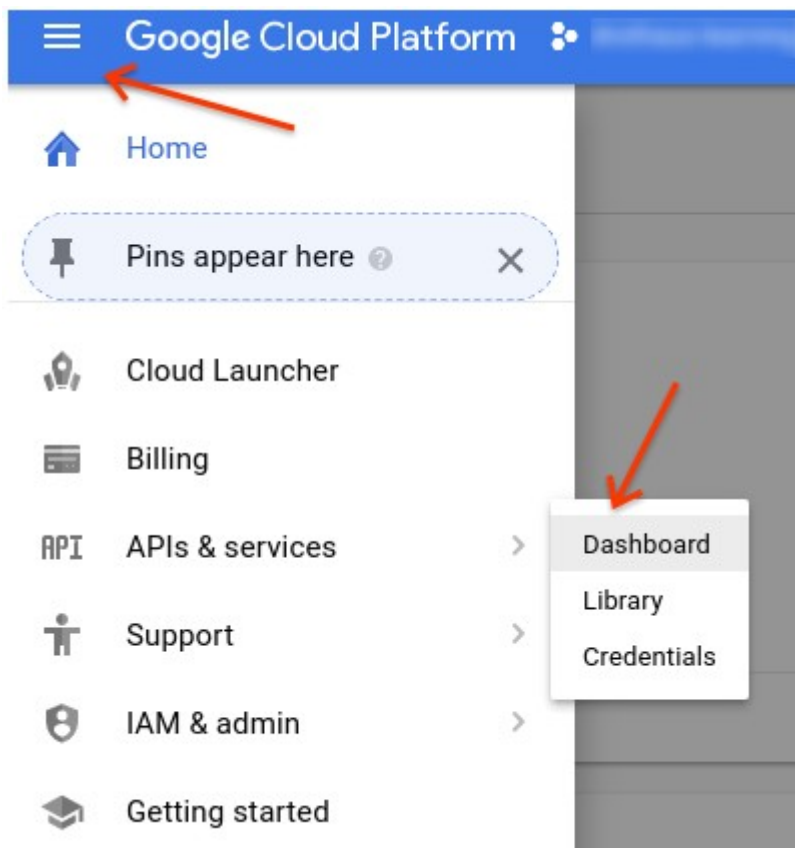
- Accept the terms and conditions.
- Do not add recovery options or two-factor authentication (because this is a temporary account).
- Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.

Enable the API

1. In the Console go to **Navigation menu** > **APIs & Services** > **Dashboard**.



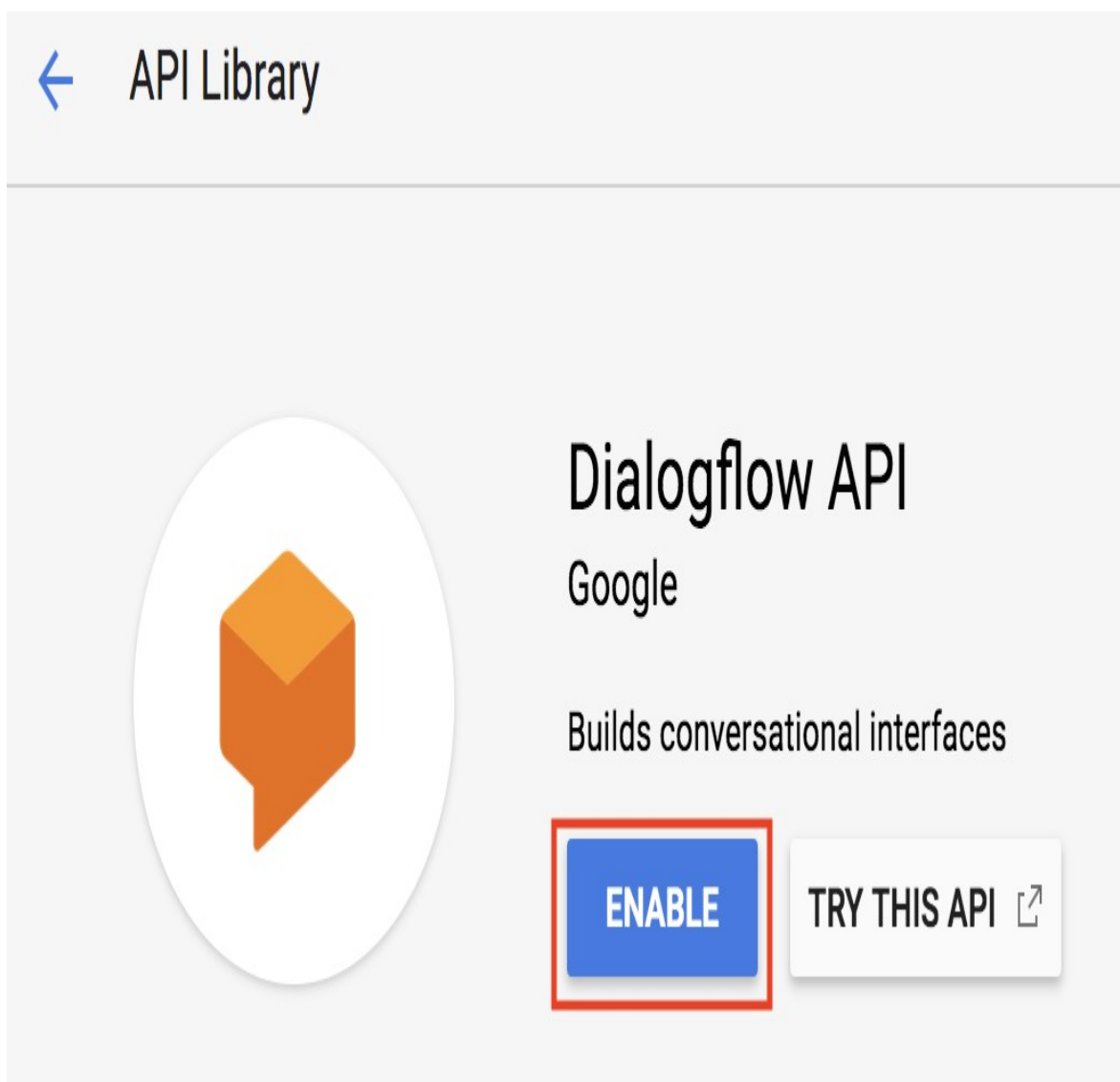
2. Click on **Enable APIs and Services**:

APIs & Services

[+ ENABLE APIS AND SERVICES](#)

3. Search for **Dialogflow**:

4. Click on the **Dialogflow API** and if the API is not Enabled, click **Enable**:



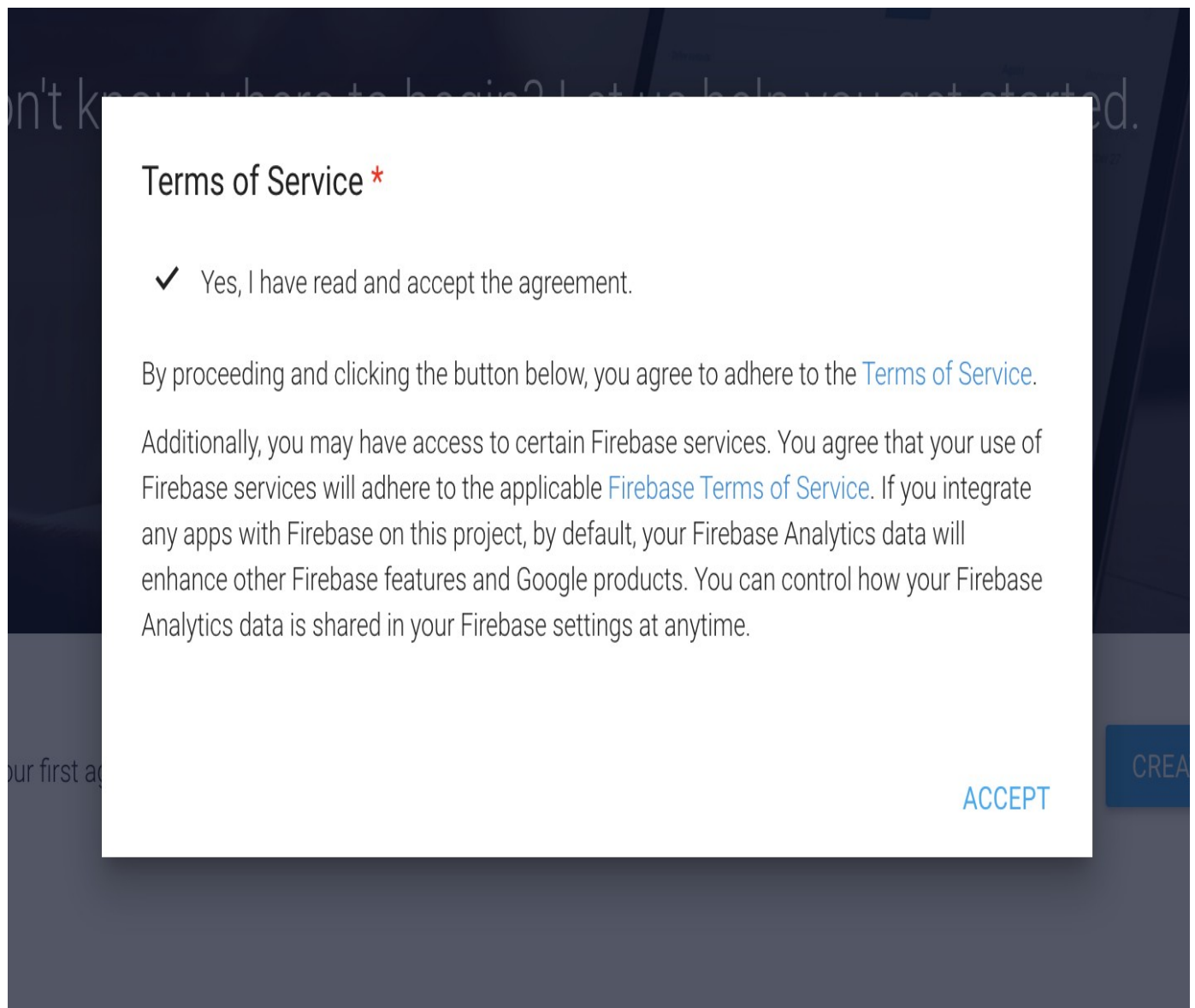
Create your first Dialogflow agent

You'll call your agent "pigeon-travel".

1. Click this link to go to the [Dialogflow Console](#).

2. If required first sign in with the Google button, and make sure to select the lab credentials you logged into this lab with. Then click **Allow**.

3. **Accept** the Terms of Service.



4. Click **Create Agent**.


5. Now add the agent information:

- Agent Name: pigeon-travel
- Default Time Zone: America/Denver
- Google Project: use your lab Project ID

6. Click **Create**.

Click Check my progress to verify the objective.

The Dialogflow console menu

You should now see the Dialogflow console. On the left is the menu panel. If you're working on a smaller screen and the menu is hidden, click on the **Navigation menu**  button in the upper left corner. The settings button takes you to the current agent's settings.

The screenshot shows the Dialogflow console interface. On the left is a sidebar with navigation options: 'pigeon-travel' (selected), 'en', 'Intents' (highlighted in blue), 'Entities', 'Knowledge [beta]', 'Fulfillment', 'Integrations', 'Training', and 'Validation'. The main area is titled 'Intents' and features a 'CREATE INTENT' button. Below this is a search bar labeled 'Search intents'. A list of intents is displayed, including 'Default Fallback Intent' and 'Default Welcome Intent'. A message states: 'No regular intents yet. [Create the first one.](#)' Below this, an explanatory text says: 'Intents are mappings between a user's queries and actions fulfilled by your software. [Read more here.](#)' At the bottom, it says: 'Before you start, check out [Prebuilt Agents](#), a collection of agents developed by the Dialogflow team.' On the right side, there is a 'Try it now' button with a microphone icon and a note: 'Please use test console above to try a sentence.' Below this is a link: 'See how it works in Google Assistant.'

The middle of the page will show the list of intents for the agent. By default, Dialogflow agents start with two intents. Your agent matches **Default Fallback Intent** when it doesn't understand what your users say. The **Default Welcome Intent** greets your users. These can be altered to customize the experience.



Intents

CREATE INTENT



Search intents



Default Fallback Intent




Default Welcome Intent

On the right is the Dialogflow simulator. This lets you test out your agent by speaking or typing messages.

Try it now



 Please use test console above to try a sentence.



See how it works in [Google Assistant](#). 

Querying your agent

Agents can be described as NLU (Natural Language Understanding) modules. These can be included in your app, product, or service and transform natural user requests into actionable data.

Try out your agent

1. In the simulator on the right, click into the text field that says **Try it now**, type "Hi", and press **Enter**.



See how it works in [Google Assistant](#). [↗](#)

Agent

USER SAYS

[COPY CURL](#)

hi



DEFAULT RESPONSE



[PLAY](#)

Greetings! How can I assist?

INTENT

[Default Welcome Intent](#)

ACTION

input.welcome

DIAGNOSTIC INFO


You just spoke to your Dialogflow agent! The **Default Welcome Intent** is preconfigured to understand basic greetings and to reply with a generic response. Note: If you didn't see the default welcome response, click **Save** to trigger the agent training and try entering "Hi" again.


Now replace the generic greeting with something that lets your users know who you are.


2. In the **Responses** section of the **Default Welcome Intent**, remove all the predefined responses like **Good Day!** and **Greetings!** and add the following responses:

- Welcome to Pigeon Travel. I can assist you with making a reservation or modifying a reservation. Which would you like ?
- Hello there. I can assist you with making a reservation or modifying a reservation. How may I help you today ?

3. Click **Save**.

Responses 

DEFAULT 

Text Response		
1	Welcome to our Pigeon Travel. I can assist you with making a reservation or modifying current reservations. Which would you like ?	
2	Hello there. I can assist you with making a reservation or modifying current reservations. How may I help you today ?	
3	Enter a text response variant	

ADD RESPONSES

4. Try it out by entering "Hi" or "Hello" in the agent simulator and see how it responds.


In the same way, you may customize the responses for your **Default Fallback Intent**.


Besides the ones provided by default, you will need custom intents that will answer specific queries for the use case. So, create your own intent.

Create your first intent

Dialogflow uses intents to categorize a user's intentions. Intents have Training Phrases, which are examples of what a user might say to your agent. For example, someone wanting to change their name on a reservation "I want to change name on my itinerary", "I ned to change my name on the booking" or "Can I change name on my reservation?". All of these queries are unique but have the same intention: to change their name on a reservation.

To cover this query, create a name.reservation intent:


1. Click on the  next to **Intents** in the left menu.
2. Add the name "name.reservation" into the **Intent name** text field.
3. In the **Training Phrases** section, click on **Add Training Phrases** and enter the following, pressing enter after each entry:
 - I want to change my name on my itinerary.
 - Can I change my name on my reservation?
 - I need to change my name on the booking.
 - I want to change my name on my hotel reservation.


Training phrases 


Search training phrases




 Add user expression

 I want to change my name on my hotel reservation

 I need to change my name on the booking

 I want to change my name on my itinerary

 Can I change my name on my reservation?

4. In the Responses section, under **Text Responses**, enter the following response in the text field: Sure I can help you to change your name on the reservation.

5. Click the **Save** button. You may also notice the messages **Agent Training started** and **Agent Training completed** on the bottom right of the screen.

This lets you know that Dialogflow is retraining your agent model based on the phrases you added.

6. Once training is done, try it out using the simulator on the right by asking the question: change name on booking?

Agent

USER SAYS

[COPY CURL](#)

change name on booking?



DEFAULT RESPONSE



Sure I can help you to change your name on the reservation.

INTENT

[name.reservation](#)

ACTION

Not available

SENTIMENT

Query Score: 0.0

DIAGNOSTIC INFO

Your agent now responds to the query correctly. Notice that even though your query was a little different from the training phrases, Dialogflow still matched the query to the right intent, which in this case is the **name.reservation** intent as shown in the simulator output.

Dialogflow uses training phrases as examples for a machine learning model to match users' queries to the correct intent. The [machine learning](#) model checks the query against every intent in the agent, gives every intent a score, and the highest-scoring intent is matched. If the highest scoring intent has a very low score, the fallback intent is matched.

Click Check my progress to verify the objective.

Extract data with entities

In this section you'll learn how to extract data from a user's query to let them change their name on the reservation. For this example, the agent will not only need to know that a user wants to change their name but also the reservation number and the new name.

Add parameters to your intents

Parameters are important and relevant words or phrases in a user's query that are extracted so that your agent can provide a proper response.

1. Click on **Intents** in the left menu.

2. Click on the **name.reservation**.

3. In the Responses, change the Text Response to: Sure I can help you to change your name on the reservation. Can I have your first name?

4. Click **SAVE**.

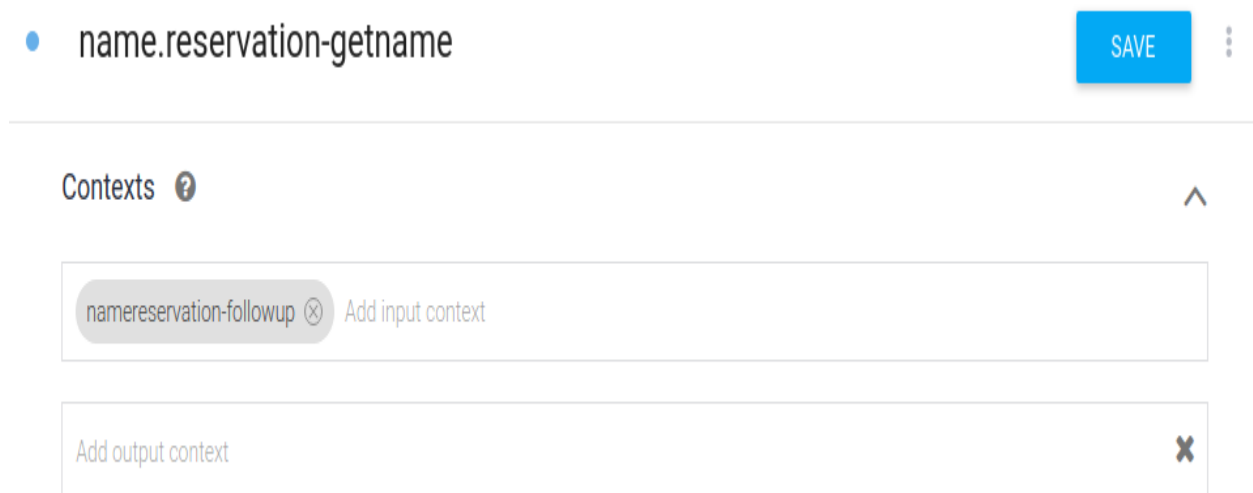
5. Click on **Intents** in the left menu.

6. Hover over the **name.reservation** intent and click **Add follow-up intent** then choose **custom**.

7. Click on the newly created follow-up intent **name.reservation - custom**.

8. Name the intent "name.reservation-getname" at the top of the intent page.

9. It should look like this



The screenshot shows the configuration page for the intent 'name.reservation-getname'. At the top, there is a blue 'SAVE' button and a vertical ellipsis menu. Below the header, the 'Contexts' section is expanded, showing two input fields. The first field contains the text 'namereservation-followup' with a close icon and the placeholder 'Add input context'. The second field is empty with the placeholder 'Add output context' and a close icon.

10. Add the following as Training Phrases:

- sam
- john
- mary

11. Under **Action and parameters**, change the **Entity** to @sys.given-name

Training phrases

Search training phrases 



” Add user expression

” mary

” john





” sam

Action and parameters



namereservation.namereservation-custom



REQUIRED 	PARAMETER NAME 	ENTITY 	VALUE	IS LIST 
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>
<input type="checkbox"/>	given-name	@sys.given-name	\$given-name	<input type="checkbox"/>

+ New parameter

12. Click **Save**.

13.The value of a parameter can be used in your responses. In this case, you can use given-name in your responses and it will be replaced with the name specified in the query to your agent.

14.In the Responses section, add the following response, then click the **Save** button: Thank you \$given-name.





15.Once the agent is done training click on **Intents** in the left menu.

16.Click on the drop down under the **name.reservation** intent.

17.It should look like this:

Intents

Search intents

-  Default Fallback Intent
-  Default Welcome Intent
-  name.reservation ^
-  ↳ name.reservation-getname

Click Check my progress to verify the objective.

Using parameter data

Try it out

Now, query your agent with change name on booking in the simulator in the right panel.

 See how it works in [Google Assistant](#). [↗](#)

Agent

USER SAYS

[COPY CURL](#)

change name on booking



DEFAULT RESPONSE



Sure I can help you to change your name on the reservation. Can I have your first name?

CONTEXTS

[RESET CONTEXTS](#)



namereservation-followup

INTENT

[name.reservation](#)

You should see the simulator asking you for your first name.

Now, in the query enter your first name and you will now have a personalized message to the user.

 See how it works in [Google Assistant](#).


Agent

USER SAYS

[COPY CURL](#)

Isabel



DEFAULT RESPONSE



Thank you Isabel.

CONTEXTS

[RESET CONTEXTS](#)

namerreservation-followup

INTENT

[name.reservation-getname](#)

ACTION

namereservation.namereservation-custom

PARAMETER

VALUE

given-name

Isabel

This is all great, but don't forget that you still need to extract other parameter values like last name and reservation number.

Use slot-filling

Although you have updated the response to include the first name parameter, how can you ensure that you get all that info from the user? The answer is slot-filling. The agent needs to make sure that if the user does not provide this info, it will need to prompt the user to provide it.

1. Click on **Intents** in the left menu.
2. Expand the drop on the **name.reservation** intent.
3. Click on the intent **name.reservation-getname**.
4. In the **Actions and parameters** enter the following parameters:

- required: true
- parameter name: reservationnumber
- entity: @sys.number
- value: \$reservationnumber
- not defined as lists
- prompts: What is your reservation number?

And then add:

- required: true
- parameter name: newname
- entity: @sys.person
- value: \$newname
- not defined as lists
- prompts: What is the new name for the reservation?

5. When completed it should look like this

Action and parameters



namereservation.namereservation-custom

REQUIRED	PARAMETER NAME	ENTITY	VALUE	IS LIST	PROMPTS
<input checked="" type="checkbox"/>	reservationnumber	@sys.number	\$reservationnumber	<input type="checkbox"/>	What is your re...
<input type="checkbox"/>	given-name	@sys.given-name	\$given-name	<input type="checkbox"/>	—
<input checked="" type="checkbox"/>	newname	@sys.person	\$newname	<input type="checkbox"/>	What is the new...
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>	—

[+ New parameter](#)

6. In the Responses section change the text response then click the **Save** button: Thank you \$given-name. I have changed the name on reservation number \$reservationnumber to be \$newname.

7. Try it out in the simulator by typing out the question: change name on booking and follow along.

Agent

USER SAYS

[COPY CURL](#)

Isabel Henriques



DEFAULT RESPONSE



Thank you Isabel. I have changed the name on reservation number 100 to be Isabel Henriques.

CONTEXTS

[RESET CONTEXTS](#)

namereservation-followup

INTENT

[name.reservation-getname](#)

ACTION

namereservation.namereservation-custom

PARAMETER

VALUE

newname

{ "name": "Isabel Henriques" }

reservationnumber

100

given-name

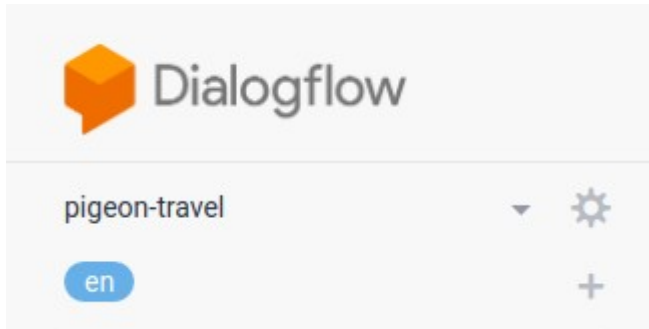
Isabel

Click [Check my progress](#) to verify the objective.

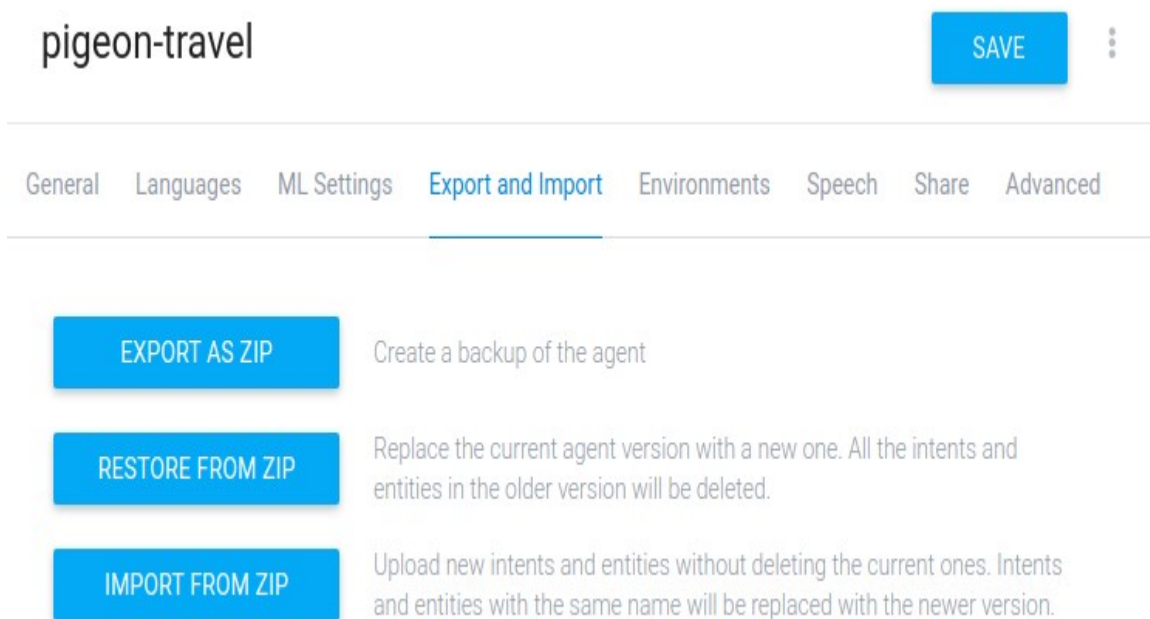
Export your agent

In this section you will export your agent as a zip file so that you can import it later when you start the next lab. This way you can reuse the intents and entities you've configured so far.

1. Click on the settings gear ⚙ icon next to your agent name in the left menu.



2. In the settings page that opens up, go to the **Export and Import** tab.



3. Click on **EXPORT AS ZIP**. This will download your agent into a local zip file.

Congratulations

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. [Our classes](#) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](#) help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated: June 17, 2021

Manual Last tested: June 17, 2021

End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied

- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Copyright 2021 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.