Dialogflow *β*

New Attempt

Due Dec 15 by 11:59pm **Points** 6 **Submitting** a file upload
File Types zip, js, and mp4 **Available** after Dec 2 at 6pm

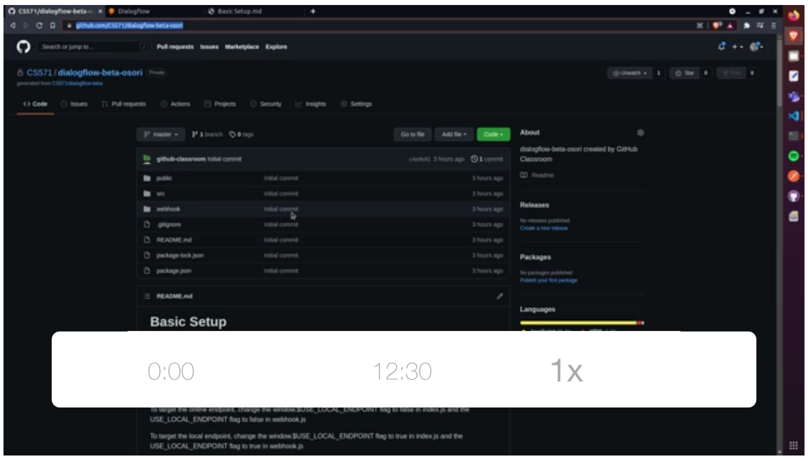
Assignment: Dialogflow β (6 Points Total)

Starter Code: GitHub Classroom Starter Code (https://classroom.github.com/a/fJwGWGf0)

This assignment is meant to teach you the basics of building a dialog-based interface. You will do this by working with **Dialogflow (https://dialogflow.cloud.google.com/)**, a software suite by Google, as well as a React app that is provided to you. You are expected to create a completely functional voice agent that also modifies the application state on the server, such that the web page is updated.

Dialogflow β Setup Tutorial

Watch the video below to get started with implementing your DialogFlow agent for WiscShop. It will walk you through how to connect the webhook with DialogFlow. Before watching the video, create a new Dialogflow agent by following the **"Create an agent" section in the official DialogFlow docs (https://cloud.google.com/dialogflow/es/docs/quick/build-agent#create-an-agent)**. You also need to be connected to the VPN.



github-classroom Initial commit

public Initial commit 3 hours ago
src Initial commit 3 hours ago
webhook Initial commit 3 hours ago
.gitignore Initial commit 3 hours ago
README.md Initial commit 3 hours ago
package-lock.json Initial commit 3 hours ago
package.json Initial commit 3 hours ago

dialogflow-beta-osori created by GitHub Classroom

README.md

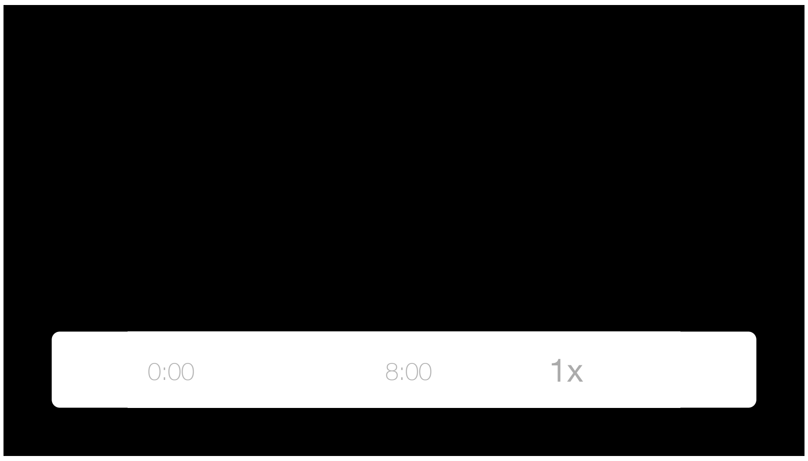
Basic Setup

To target the online endpoint, change the window.$USE_LOCAL_ENDPOINT flag to false in index.js and the USE_LOCAL_ENDPOINT flag to false in webhook.js

To target the local endpoint, change the window.$USE_LOCAL_ENDPOINT flag to true in index.js and the USE_LOCAL_ENDPOINT flag to true in webhook.js

Alternatively, read **DialogFlow Beta Setup Guide.pdf (https://canvas.wisc.edu/users/146319/files/23289734/download?verifier=XMFulpQjK6pVQaqKXxMoSbnyx6H95mpVf48wG3vs&download_frd=1)** .

Local Endpoint Server Setup Tutorial (Optional)



You can access the Docker image for the local endpoint server [here (https://github.com/CS571/WiscShopLocalServer)](https://github.com/CS571/WiscShopLocalServer). Using the local endpoint is completely optional, so use the local endpoint *only if you are experiencing a long latency* between your webhook and our online API endpoint at `http://cs571.cs.wisc.edu:5000`.

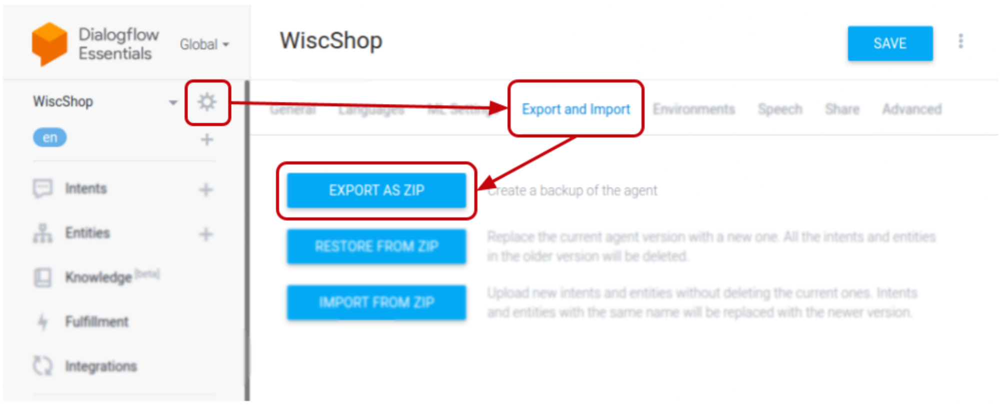
Potential Gotchas/Helpful Suggestions:

- **Start working on the project early!** Come to our office hours or make a Piazza post whenever you feel stuck.
- We highly suggest reviewing the **rubric and specification** below. Information on using the API can be found in the Dialogflow Beta project README.
- Feel free to **peruse the starter code**. It has a number of fetch examples with more sophisticated filtering of products.
- Watch the **setup tutorial** or read the **setup guide** to get a good idea of how to set everything up.
- Review the **Dialogflow** [1 (https://hci-curriculum-uwmadison.github.io/CS571/lectures/12-Build-Dialogflow-1.pdf)](https://hci-curriculum-uwmadison.github.io/CS571/lectures/12-Build-Dialogflow-1.pdf) **/**[2 (https://hci-curriculum-uwmadison.github.io/CS571/lectures/12-Build-Dialogflow-2.pdf)](https://hci-curriculum-uwmadison.github.io/CS571/lectures/12-Build-Dialogflow-2.pdf) **lectures** and refer to the [official Dialogflow docs (https://cloud.google.com/dialogflow/es/docs)](https://cloud.google.com/dialogflow/es/docs) to understand concepts and features from Dialogflow.
- If you want to see how to update the application without Dialogflow, feel free to **use** [Postman (https://www.getpostman.com/)](https://www.getpostman.com/) to do the API calls.

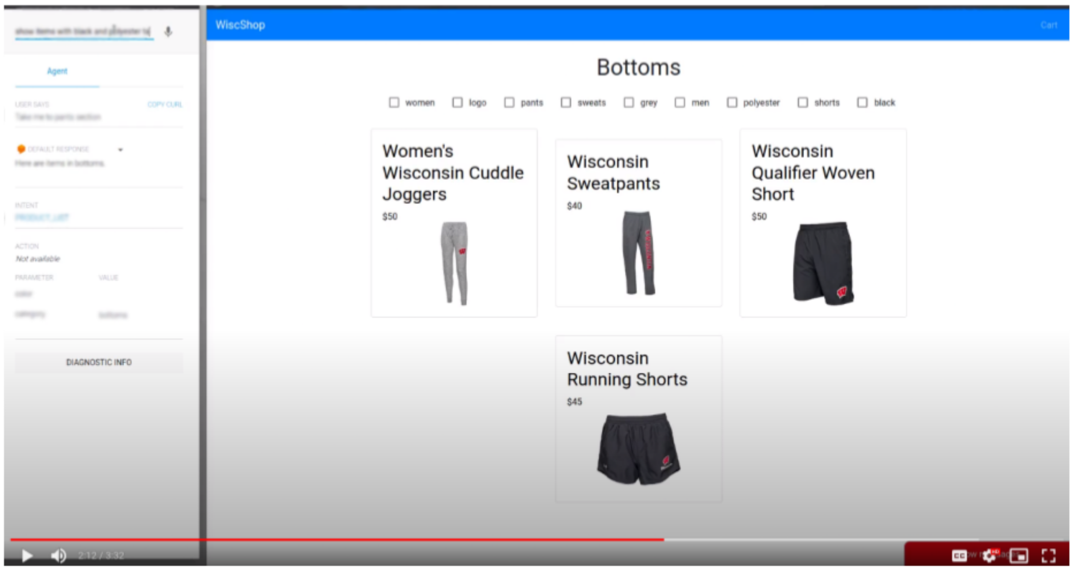
Submission Details

To submit the assignment, you will need to do **three** things.

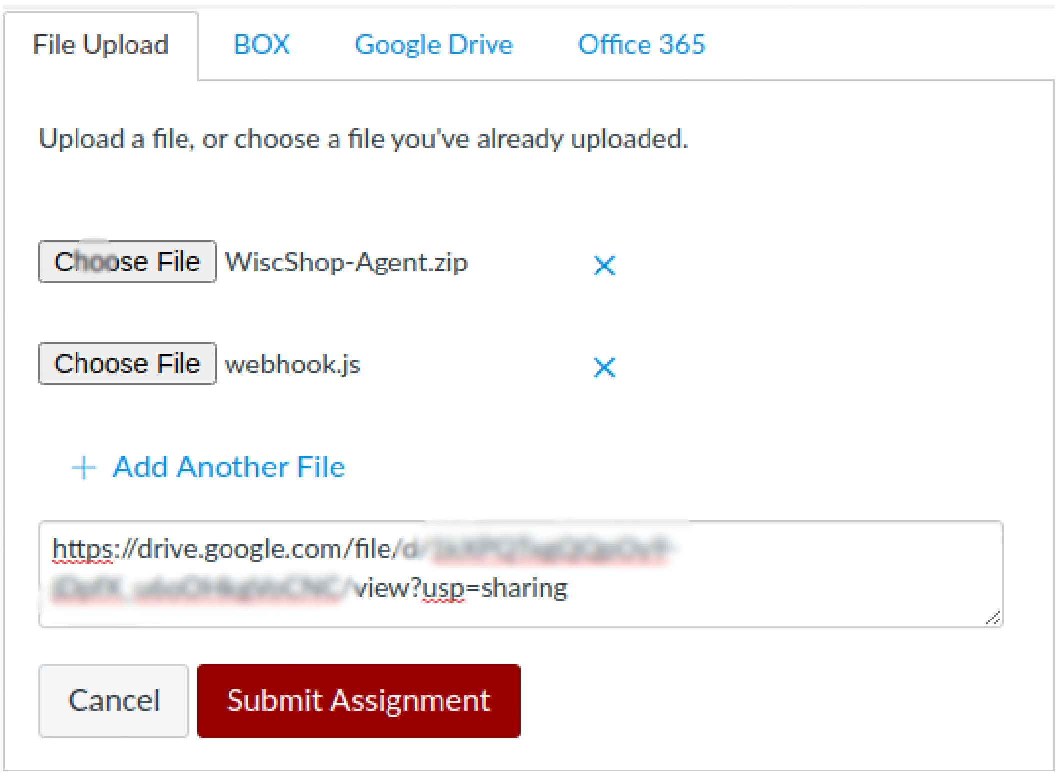
1. You will be providing the **.zip file of your exported Dialogflow agent**. You can download this file from the settings pane:


Dialogflow Essentials
Global
WiscShop
SAVE
en
Intents
Entities
Knowledge beta
Fulfillment
Integrations
General
Languages
ML Settings
Export and Import
Environments
Speech
Share
Advanced
EXPORT AS ZIP
Create a backup of the agent
RESTORE FROM ZIP
Replace the current agent version with a new one. All the intents and entities in the older version will be deleted.
IMPORT FROM ZIP
Upload new intents and entities without deleting the current ones. Intents and entities with the same name will be replaced with the newer version.

2. Secondly, you will be uploading the `webhook.js` **file** on Canvas, a starter of which is provided in your GitHub Classroom repository.
3. Finally, you will need to upload **a screen recording of you demonstrating each feature specified in the rubric** to your UW Madison Google Drive. Copy a share link(Anyone in G-suite can view) into the text submission field of your Canvas submission or include it as a submission comment. For your demonstration, you should have a window with the WiscShop interface open next to a window with the Dialogflow console.



Your Canvas submission should look like below:



File Upload | BOX | Google Drive | Office 365

Upload a file, or choose a file you've already uploaded.

Choose File WiscShop-Agent.zip ×

Choose File webhook.js ×

+ Add Another File

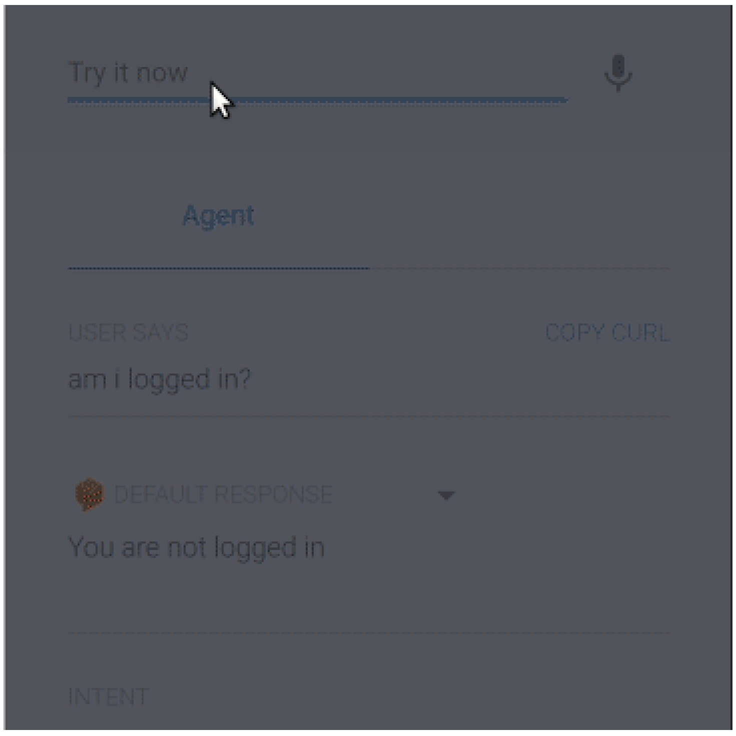
https://drive.google.com/file/d/[redacted]/view?usp=sharing

Cancel | Submit Assignment

Specification

IMPORTANT: You must enable webhook for intents that are used to implement the following features. No points will be given if webhook is not utilized (i.e. Don't hard code responses in the Responses section of the Dialogflow console when they are supposed to come from interacting with the API).

- **Login**



Try it now

Agent

USER SAYS | COPY CURL

am i logged in?

DEFAULT RESPONSE

You are not logged in

INTENT

  - User is able to login with username and password. You do not need to handle account creation.

- *Note*: User should manually log in to the WiscShop interface; the shopping agent does not need to navigate the user from the login page to the landing page.

- **Queries**
    - *Categories:* User should be able to query about the types of products offered.
    - *Tags:* User should be able to inquire about the types of tags for a specific category.
    - *Cart:* User should be able to request information about what is in their cart (e.g. total number and type of items, total cost, etc.).
    - *Product Info:* User should be able to request information about a product. If the product has reviews, they should be able to inquire about reviews and average ratings.
- **Actions**
    - *Tags:* User should be able to narrow down the search results within a category by specifying tags, e.g. "Show me all the red ones".
    - *Cart:* User should be able to add/remove items (or multiple of an item) to/from your cart. They should also be able to clear their cart.
    - *Cart Confirm:* User should be able to review their cart and then confirm their order in the cart. (When user asks for a review, they should be navigated to either the cart page or cart-review page. When user would like to confirm the purchase, they should be navigated to the cart-confirmed page.)
- **Navigation**
    - User should be able to navigate through the application with the voice assistant using natural language, e.g., "Take me to the home page" or "Show me the hats".
    - For a full breakdown of the various routes in the application, see the WiscShop readme.
- **Messages**
    - Messages should be updated when the user or agent says something.
    - Upon starting a new session, the messages should be cleared.
    - *Note*: You do not need to show messages before the user is logged in.
- **Design and Personality**
    - Users should be able to converse with the agent in natural language to perform tasks.
    - You shouldn't need to specify product id's to navigate.
    - The agent should have a personality, and should engage in expected turn-taking behavior.

Dialogflow beta

Criteria	Ratings		Pts
Login User is able to login with the voice assistant.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Queries: Categories User is able to request information about categories.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Queries: Tags User is able to request information about tags for a specific category.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Queries: Cart User is able to request information about their cart (e.g. total number/type of items, total, etc).	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Queries: Product Info User is able to request information about a product. If the product has reviews, they should be able to inquire about reviews and average ratings.	**1 to >0.0 pts Full Marks**	**0 pts No Marks**	1 pts
Actions: Tags User should be able to narrow down the number of items shown based on tags.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Actions: Cart User should be able add/remove items (or multiple of an item) to/from their cart.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Actions: Cart Confirm Users should be able to review, then confirm their cart.	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts
Navigation: Pages Users are able to navigate throughout the application with the voice assistant using natural language, e.g. "Take me to the home page" or "Show me the hats".	**0.5 to >0.0 pts Full Marks**	**0 pts No Marks**	0.5 pts

Criteria	Ratings		Pts
Messages Messages should be updated when the user or agent says something. Upon starting a new session, the messages should be cleared.	**0.5 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	0.5 pts
Design and Personality Users should be able to converse with the agent in natural language to perform tasks. In other words, you shouldn't have to specify product id's.	**0.5 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	0.5 pts
			Total Points: 6