CS-639 Building User Interfaces, Fall 2020, Professor Mutlu

Dialogflow α (3 Points)

Experience Prototyping Conversational Interactions

In this assignment, you will start your work toward designing and developing your Module 3 deliverable. We discussed in class that designing conversational interfaces has unique challenges and that ideation and prototyping methods that work very well in other design problems do not work well here. The good news is that we are also subject matter experts in conversation, but the bad news is that our expertise is encoded in our brains and is not readily available for us to use, what we called *tacit knowledge*. This is where *experience prototyping* comes into the picture: by simulating the social and/or the physical setting for the interaction and acting out the interactions using methods such as *bodystorming*, we unlock our expert knowledge and apply it to the design problem.

In this assignment, you will engage in experience prototyping for a *conversational shopping assistant*, which will serve as the basis for developing the intents and entities for the first prototype of your Dialogflow implementation. Specifically, the Dialogflow β tasks provided below should inform the development of the scenarios in Part 1, and the bodystorming of these tasks in Part 1 should be the basis for the specification of the intents, entities, and responses in Part 2.

Part 1. Experience Prototyping (1.5 Points). The first part of the assignment will involve engaging in *bodystorming* to generate ideas and specifications for your shopping assistant.

Part 2. Agent Specification (1.5 points). In the second part of the assignment, you will build on the outcome of your experience prototyping activity to develop specifications for the agent you will build in Dialogflow.

Submission Details

You will submit a completed version of this document in PDF format to Canvas.

Part 1. Experience Prototyping (1.5 Points). In this step, you will follow a process very similar to the process we followed for the in-class activity on experience prototyping, paying particular attention to *bodystorming* for idea generation. In the context of designing a shopping assistant robot, follow the steps below:

1. *Define context* — This is given to you: users interacting with a conversational shopping assistant embedded within a clothing retail website. There is no deliverable for this step.
2. *Develop scenarios* — Think about how the shopping assistant will help users. What are some tasks the shopping assistant can help users with? Develop 3 scenarios. The tasks from Dialogflow β (provided below) should be the basis of these scenarios. Reviewing the WiscShop API readme will also be helpful in developing your scenarios.
3. *Identify design goals* — Determine what the shopping assistants can do to assist in these tasks. Consider aspects of the task where the assistant can bring added value. Our goal is not designing a fully autonomous assistant that could take care of everything with minimal input from the user, but what is called a *mixed-initiative design* where the assistant does what it's good at and the user does what the user is good at.
4. *Setup environment* — You can use the retail store provided with Module 3 starter code and/or another clothing retail store as your environment or prop during your acting.
5. *Act out interaction* — Ask a friend, family member, or another student in class to help you bodystorm user interactions with the shopping assistant to develop ideas and to more concretely define user and system behavior and interactions with the environment. Act out and record a transcript on at least one interaction for each scenario.
6. *Develop insight* — Capture the conversations from your bodystorming session and any other insight you have gained from the previous step in notes and translate them into a flowchart representation of the interaction.

Tasks that your Dialogflow β agent should support are listed below:

- Login
    - User is able to login with username and password. You do not need to handle account creation.
    - NOTE: It is sufficient if the user enters this information as a text query (typing), in case the username and/or password is hard to parse. It should still be english, e.g. "Log in with username <username> and password <password>."
- 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, then confirm their cart.
- 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.

Your deliverables will be the scenarios and design goals you have focused on, the transcripts of the bodystorming sessions, and a flowchart representation of the conversational capabilities suggested by your experience prototyping through your 3 scenarios. Your flowcharts can be in the form of a graph where the nodes are system behaviors and arrows are user behaviors. To generate flowcharts, you can use SmartDraw (using your NetID login) or free versions of other tools, such as LucidChart or Creatly.

Scenario and design goals:

1. Log In
   Design goals:
    - Agent understand the user's request to log in, and able delivers the user to the login page
    - Log into the user's account for the user according to the username and password user specified
    - Confirms the user that they have logged in successfully, else notify them with error message in natural language
2. Search for a plush with the highest review and then the lowest price to add to cart.
   Design goals:
    - Navigate to the <Category> specified by user
    - In <category>, find the item(s) of the desired ratings, and display to user.
    - Find the item(s) of the desired price within the list of items with desired review.
    - Display the item(s) in the filtered list to the user and allow user to chose the one(s) they would like to add to cart.
    - Add the items specified by the user to the cart.
3. Delete the a most expensive item from cart
   Design goal:
    - Find the item(s) of the desired price in cart and display to user
    - Ask user which one(s) they would like to delete from cart
    - Delete the items specified by the user.

Bodystorming transcripts

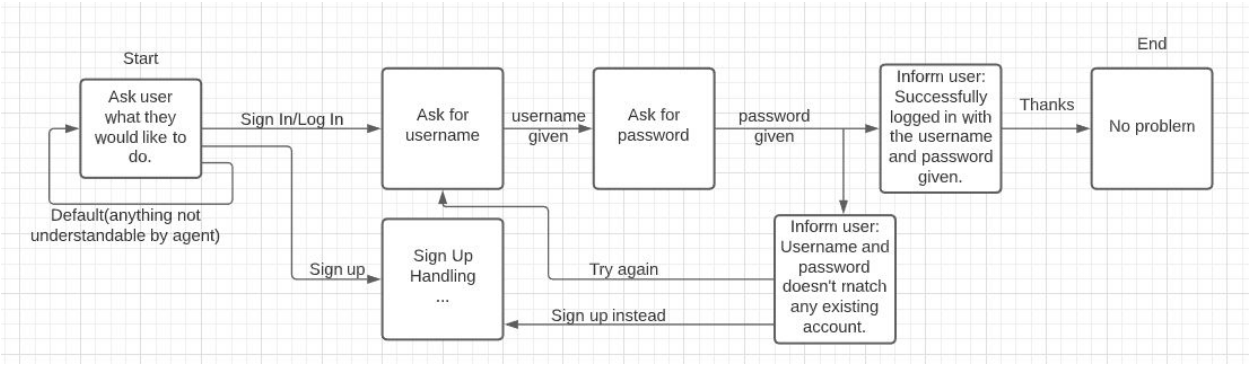
1. Log In
    - Agent: Welcome to WiscShop! What would you like to do today?
    - User: I want to log in.
    - Agent: Alright, what is your username?
    - User: mtan_te

- Agent: Got it! What is your password?
- User: password
- Agent: Great. Logging you in…
- Agent: You are in your account! Welcome, mtan_te!
- User: Cool, thanks.
- Agent: You are welcome.

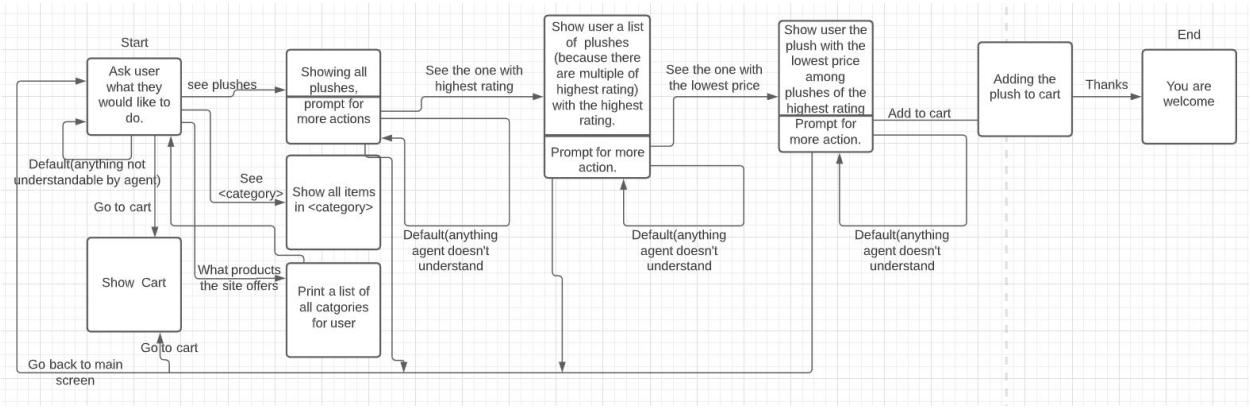
2. Search for a plush with the highest review and then the lowest price to add to cart.
    - User: I want to see the plushes.
    - Agent: (Navigated to the plushes screen) Got it. Now showing plushes.
    - User: Show me the ones with the highest rating.
    - Agent: We have Bucky Badger Plush, Gameday Bucky Plush, and Bucky Badger Pillow, all with 5 stars reviews.
    - User: Which one is the cheapest one?
    - Agent: (Navigated to the page with Bucky Badger Plush) Bucky Badger Plush is the cheapest, with a price of 13 dollars.
    - User: I'll take this one.
    - Agent: (Added Bucky Badger Plush to Cart) Alright, I have added Bucky Badger to cart for you!
    - User: Thanks.
    - Agent: No problem!
3. Delete the most expensive item from cart
    - User: What's the most expensive item in my cart?
    - Agent: Well, the most expensive items are Women's Wisconsin Cuddle Joggers and Wisconsin Qualifier Woven Short, both with the price of 50 dollars.
    - User: Delete the shorts for me.
    - Agent: (Deleted the Wisconsin Qualifier Woven Short in Cart) Okay, deleted the Wisconsin Qualifier Woven Short.
    - User: Great. Thank you.
    - Agent: My pleasure.

Flowcharts

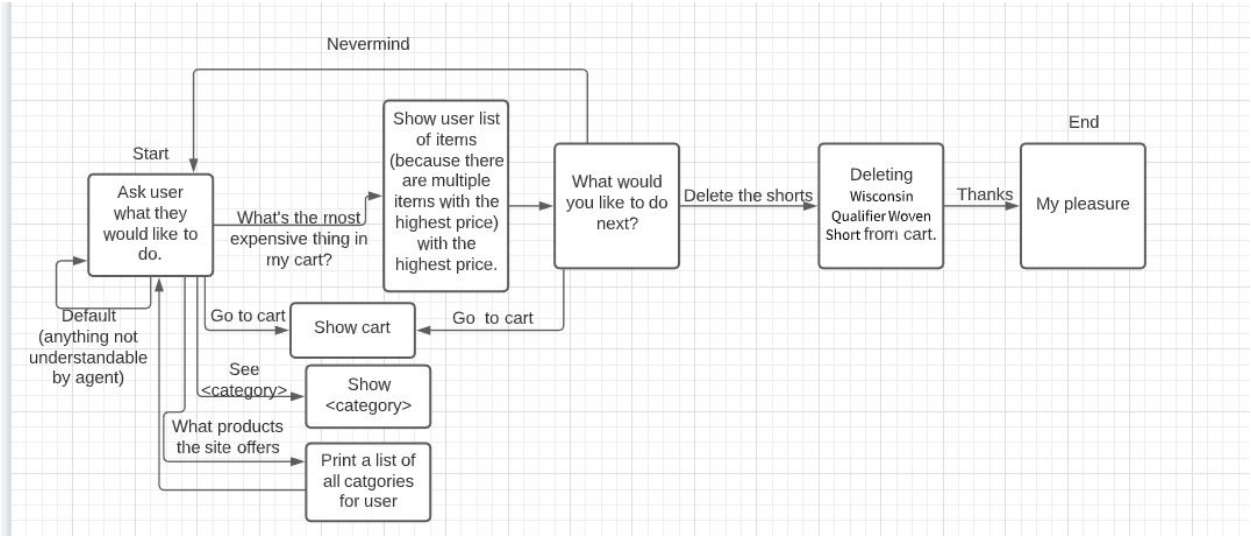
1. Log in



2. Search for a plush with the highest review and then the lowest price to add to cart.



3. Delete the most expensive item from cart



Part 2. Agent Specification (1.5 Points). In this step, you will apply what you learned in your experience prototyping activity to the design of the agent you will be creating. More specifically, you will draw on the outcome of your bodystorming session to determine the *intents* and *entities* that your agent will utilize in its conversation, and consider how you will use them and server data to provide responses.

If a particular intent or response is infeasible to implement the way you imagined in your bodystorming session, explain why, and propose an intent or family of intents which can be realistically implemented using the Dialogflow framework that will support the same functionality.

In this part, you will provide three main deliverables:

1. A list of all *intents* you will use (provide 10 training examples for each intent).
2. A list of all *entities* (provide at least five examples for each entity) you will be using with your agent.
3. For each *intent*, develop agent responses, specifically what it will *say in reply* (at least three responses to avoid repetition) and what it will *do* to change the GUI.

For a full description of what the GUI can do, and the requirements of the agent, see the Dialogflow β assignment details and the WiscShop API readme.

Entities:

- Category
  Examples:
    - Hats
    - Tees
    - Leggings
    - Bottoms
    - Sweatshirts
- Price
  Examples:
    - Under 20 dollars
    - Around 50 dollars
    - Most Expensive
    - Cheapest
    - 30 dollars
- Review
    - Best
    - Lowest
    - 5 star
    - At least 4 stars
    - Highest
- Item name
    - Bucky Crew Neck Sweatshirt
    - Wisconsin Leggings
    - Bucky Badger Pillow
    - 150 Year Commemorative Hoodie
    - Wisconsin Sweatpants
- Tag
    - Red
    - Women
    - Logo
    - White
    - cotton

Intents:

- Sign up
  Training phrases:
    - I would like to sign up
    - I would like to create a new account
    - I want to sign up

    - Sign up
    - Help me sign up
    - I wish to sign up
    - Sign up, please
    - I need to sign up
    - Sign up a new account
    - Create a new account for me
    - Sign up for me

  Responses:
    - Sure, would you fill out your information here?
    - Got it. Would you provide me with your information on the screen?
    - Alright. What would be your name, username, and password?

  Action:
    - POST request to the /users endpoint to get a token for accessing user information
    - Move to /username screen

- Log in

  Training phrases:
    - I would like to log in
    - I would like to sign in
    - I want to log in
    - Log in
    - Help me log in
    - I wish to sign in
    - Log in, please
    - I need to log in
    - Sign into my account
    - Sign me in
    - Log in for me

  Responses:
    - Sure, may I have your username and password?
    - Got it. Would you provide me with your username and password?
    - Alright. What would be your username and password?

  Action:
    - GET request to the API for the token to access user information
    - Move to /username screen

- Navigate by category

  Training phrases:
    - I want to see <category>
    - Show me <category>
    - What's in <category>
    - <category>
    - I would like to visit <category>
    - Visit <category>

- Go to <category>
- Navigate to <category>
- <category> page
- Show <category>

Response:

- Okay, navigating to <category> page.
- Got it. Now showing <category>.
- Alright, here are the items in <category>.

Action:

- Move to /username/<category> page to show all products within the category specified

- Navigate by tag(could have more than one tag)

  Training phrases:

  - Show me the <tag> ones
  - I only want to see <tag> and <tag> ones
  - Check the <tag> box for me
  - <tag>
  - Which ones are <tag>?
  - I'm thinking <tag> and <tag> ones
  - Just the <tag> ones
  - Show <tag>
  - Just the <tag> ones should be good
  - <tag>, please
  - Select <tag> and <tag>

  Response:

  - Here are the <tag> ones.
  - Showing just the <tag> ones.
  - These are the <tag> ones.

  Action:

  - Select the tag(s) specified by the user and show only the items with the tags.
  - Possibly print the names of the items into the chat box between user and agents.

- Filter by review(for each product by average)

  Training phrases:

  - I only want to see the ones with <review>
  - Which ones have <review> reviews
  - Show me the ones with <review>
  - Can I see the <review> ones?
  - Just the ones with <review>, please
  - What do you have <review> reviews?
  - Show me the <review> products
  - I would like to view <review> ones
  - The <review> ones would be nice to see, thanks
  - Can you show me which ones have <review> reviews?

  Response:

- Here are the items with <review> reviews: ……..(names of products with desired reviews)
- We have ……. (names of products with desired reviews) For the reviews you are looking for.
- There are x(number of products) products with the review you asked for, and they are …….(names of products with desired reviews).

Action:

- Store the items with the desired reviews(by average) in a matched list, show only the items in the list if this can be achieved by GUI, else print the names of the items with the desired reviews to the chat to show the user.

- Filter by price

  Training phrases:

  - I only want to see the ones with prices of <price>
  - Which ones are <price>?
  - Show me the ones that are <price>
  - Can I see the <price> ones?
  - Just the ones that are <price>, please
  - What do you have that are <price>?
  - Show me the <review> products
  - I would like to view the <price> ones
  - The <price> ones would be nice to see, thanks
  - Can you show me which ones are <price>?

  Response:

  - Here are the items that are <price>: …….(names of products with desired price)
  - We have ……. (names of products with desired price) For the price you are looking for.
  - There are x(number of products) products with the price you asked for, and they are …….(names of products with desired price).

  Action:

  - Store the items with the desired price(either direct price or relative prices, such has the cheapest, or the most expensive) in a matched list, show only the items in the list if this can be achieved by GUI, else print the names of the items with the desired price to the chat to show the user.

- Add to cart

  Training phrases:

  - Add this to the cart.
  - I want this one.
  - I would like the <tag> one.
  - I want to have the <tag> one in my cart.
  - Buy the <tag> one.
  - I'll have the <tag> one.
  - I need the <price> one.
  - I would like the <price> one.
  - I want to purchase the <review> one
  - Can I have the <review> one?

Response:

- Okay. Successfully added <item name> to cart.
- Sure, I have added it to the cart for you.
- Alright, <item name> is in your cart!

Action:

- Show the user a message confirming that the item has been added to cart.
- If at a specific item page, the alert message(added) should show, else nothing would appear on the GUI.
- A PUT request is sent to update the user's cart information

- Remove from cart

Training phrases:

- I don't want <item name> anymore.
- Delete <item name>
- Remove the <price> one from my cart
- <price> is too high, I don't want it anymore.
- Remove the <review> one.
- Take away the <item name>
- Delete the one with <review>
- Clear my cart, please.
- Take away <item name>
- Have <item name> deleted, thanks.

Response:

- Okay. Deleting <item name> for you.
- Got it. Removing <item name>.
- Alright, I have deleted <item name>

Action:

- Show the user a message confirming that the item has been deleted from cart.
- No additional thing would appear on the GUI.
- A DELETE request is sent to update the user's cart information

- Go to cart

Training phrases:

- Get me to my cart.
- I want to see my cart.
- Show me my cart.
- Let me see my cart.
- Cart.
- I need to see my cart.
- I would like to checkout.
- I want to view my cart, please.
- Can I take a look at my cart?
- Navigate me to my cart, please.

Response:

- Showing you your cart now.

    - Here is your cart.
    - Alright, taking you to your cart.

  Action:
    - Show the user a message confirming that the current page is cart.
    - GUI moves to the Cart screen that belongs to the user.
- Go to main page

  Training phrases:
    - Home page.
    - Main page, please.
    - I want to go back to home.
    - I would like to go to the main page.
    - I want to reselect a category.
    - Bring me to the home page.
    - Can I see the main page, please?
    - I need to take a look at the main page.
    - Show me the main page.
    - Let me see the main page.
    - Get me to the main page.

  Response:
    - Here is the main page, you can select categories or navigate to the cart here.
    - At the main page now.
    - You arrived at the main page.

  Action:
    - Show the user a message confirming that the current page is Wiscshop homepage.
    - GUI moves to the Main screen that shows all categories under the /username endpoint.

(These are simpler ones with less possible training phrases I could think of)

- See Categories

  Training phrases:
    - What categories do you have?
    - What are the categories?
    - Show me all the categories.

  Response:
    - We have hats, sweatshirts, plushes, leggings, tees, bottoms.
    - Our categories are hats, sweatshirts, plushes, leggings, tees, bottoms.
    - There are 6 categories, and they are hats, sweatshirts, plushes, leggings, tees, bottoms.

  Action:
    - Display all categories on the screen of the chat box
- See tags

  Training phrases:
    - Show me all the tags.
    - What tags are in <category>
    - Show me the tags in <category>

- What tags are there?
- Which tags do you have?

Response:

- We have …(tag names within the current category or all tags if no category).
- The tags are …(tag names within the current category or all tags if no category).
- There are x(number of tags) tags, and they are…(tag names within the current category or all tags if no category).

Action:

- Display all tags within the current categories(or all tags if no category is chosen) on the screen of the chat box