Auto-Generated Client Documentation

https://kachhabra02.github.io/CSCE-482-Concierge

Hand-Written Client Documentation

BellButton.jsx

BellButton:

- Input: None
- Output: Button and text displayed on the MapScreen
- Description: When this button is clicked, it will bring back the user to the chatbot screen and play a bell sound.
- Used by: MapScreen()

ChatBox.jsx

ChatBox:

- Input: React State variables for the selected city, user preference vector, and message history.
- Output: The chatbot display which includes the robot picture, message box, and message history.
- Description: Displays a chat interface with a virtual concierge for restaurant recommendations
- Used by: Chatbot()

Dollar.jsx

Dollar:

- Input: Array representing the price range
- Output: Filled or outlined dollar icons representing the price range.
- Description: Displays dollar signs based on the price range
- Used by: RestCard()

Star.jsx

Star:

- Input: The numerical star rating value and number of reviews.

- Output: Filled, outlined, or half-filled dollar icons representing the ratings along with the number of reviews.
- Description: Represents a star rating component with optional half-star and review count
- Used by: RestCard()

ResultButton.jsx

ResultButton:

- Input: None
- Output: A button with an image used for navigation to the map screen.
- Description: The button component to navigate to the map screen from the chatbot screen.
- Used by: ChatBox()

RestCard.jsx

RestCard:

- Input: This includes restaurant name, star rating, number of reviews, an array of cuisines, restaurant address, opening hours, additional attributes, image URL, the total number of images, phone number, yelp URL, restaurant rank, and the State component of the highlighted card.
- Output: A card with all the input, displayed on the Map Screen's carousel.
- Description: Displays detailed information about a restaurant
- Used by: CardScreen()

CardScreen.js

CardScreen:

- Input: The restaurant JSON data, the State component for the selected card, and the forceUpdate flag.
- Output: A carousel of cards for each restaurant in the JSON.
- Description: Displays a RestCard component for each restaurant in the carousel.
- Used by: MapScreen()

Chatbot.js

Chatbot:

- Input: The state component for the selected city, user preference vector, and message history.
- Output: Calls the ChatBox component.
- Description: This function calls the ChatBox component and passes all the parameters.
- Used by: App.js

MapScreen.js

MapScreen:

- Input: The selected city and user preferences vector.
- Output: A page with a map and a carousel of cards.
- Description: This page has a map with pins and calls the CardScreen component.
- Used by: App.js

StartScreen.js

StartScreen:

- Input: None
- Output: Page with title and a bell image button.
- Description: The home page of our application which navigates to the map screen.
- Used by: App.js

Auto-Generated Server Documentation

https://my-concierge-server.vercel.app/api/docs

Hand-Written Server Documentation

nlp.py

getChatResponse:

- Input: Integer scores for each of the categories used for matching

- Output: Tailored response to send back to the user
- Description: Used to make the chatbot seem more lively. Tries to build a response to the user's previous message that fits what kind of food they are looking for.
- Used by: getUserPreferenceVector()

getUserPreferenceVector:

- Input: User message and previous user preference vector
- Output: Updated user preference vector based on the new message
- Description: Uses a predefined query to first expand the new message to get more details and then a large predefined query to extract features. Updates the user preference vector with the new findings.
- Used by: prompt()
- Uses: getChatResponse()

recommender.py

getRecommendations:

- Input: User's city and the current user preference vector
- Output: Top recommended restaurants
- Description: Queries the database to get all restaurants in the city and builds a matrix of their attributes; then performs a dot product between the matrix and the user preference vector to get the best matches.
- Used by: recommendation()
- Uses: getRestaurants(), getCategories(), getAttributes()

database.py

getCategories:

- Input: None
- Output: Returns all the different restaurant categories from the database
- Description: Executes sql command to retrieve the categories.
- Used by: getRecommendations()

getAttributes:

- Input: None
- Output: Returns all the different restaurant attributes from the database
- Description: Executes sql command to retrieve the attributes.
- Used by: getRecommendations()

getRestaurants:

- Input: Selected city
- Output: All restaurants within the selected city
- Description: Executes sql to get all the restaurants and formats the data.
- Used by: getRecommendations()