

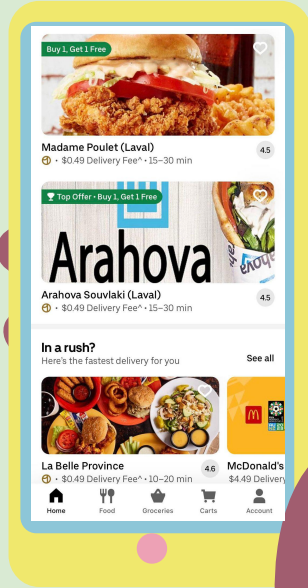
EATBOT: THE UBER EATS CHATBOT

TEAM 8

Keani Schuller
Alisa Liu
Freddy Chen
Tashfeen Ahmed



UBER EATS



- The food delivery service Uber Eats is a division of Uber Technologies Inc.
- Users can use Uber Eats via smartphone app or website, where they can place meal orders from a variety of restaurants and food vendors for pick up or delivery.
- The online platform offers a variety of cuisines and eating alternatives, and operates in many locations throughout the globe.
- Customers have the option to personalize their orders, explore menus, and follow their deliveries in real time.

OUR RATIONALE



- EatBot aims to improve consumers' entire experience by offering enhanced functionalities and meal delivery services on their mobile and desktop application.
- Target Audience: all the users of food delivery services
- EatBot will assist users while browsing menus, helping with their eating habits through data analysis and providing better customer support.
- This will be offered through enhanced recommendations as well as integrating new functionalities to create competitive advantage in the food delivery service industry.

SCENARIOS CONSIDERED: REPLICATED

- Facilitated Order Placement
 - Order Creation & Customization: add items to a user's cart for them based on what they want to eat while handling order customization such as adding side dishes or choosing spice levels
 - Payments: It can integrate UberEats' secure payment system for easy transactions
- Enhanced Customer Support
 - FAQ: answer frequently asked questions about restaurants, menus, deliveries, payments, etc.
 - Feedback: collect feedback on the food and delivery experience from users after a completed order and register issues quicker than their current system to relate them to a support employee when needed.
 - User Reports: It can use data such as the total amount spent over a specified time period or top restaurants based on their order history.



SCENARIOS CONSIDERED: NEW

- Search and Browsing Enhancement
 - Image recognition: EatBot can recognize dishes from photos uploaded
 - Surprise meal: when the user doesn't know what to eat, the chatbot can surprise users with a random restaurant and dish
- Recommendations and Meal Planning
 - Recommendations: provide personalized recommendations based on preferences, moods, and so on
 - Suggestions: provide day/weekly meal plan based on certain criteria, such as user budget, cuisine type, calorie intake, etc.
- EatsWrapped
 - Creating visualizations demonstrating interesting customer information



So, what did we select?

LEVERAGING NEW FUNCTIONALITIES

We decided to capitalize on enhancing current user experience by improving recommendations and adding new features within EatBot.

Image Recognition

User can upload or provide URL of a image and the chatbot will match it to a dish on the application

Surprise Food Box

Creation of a complete surprise meal selection by the chatbot



Recommendation & Meal Plans

Added feature based on budget, occasion, etc.

EatsWrapped

Using customer data to create fun, shareable infographics

PYTHON LIBRARIES AND APIs



Pandas

Used pandas to create dataframes of the data ingested by the chatbot to process the data easier



Matplotlib

Creating visualizations for EatsWrapped, demonstrating the data analysis done for the user



Clarifai API

Connecting to general-image-recognition & food-item-recognition model to identify dishes

FUTURE CODE EXTENSION

- We can add further functionalities to ask for customers' feedback on how they think of our Chatbot services.
 - For example, if our information filter is accurate enough to meet customers' satisfaction
- Consider precise location of users to calculate expected delivery time



HOW EATBOT WILL BENEFIT UBEREATS

- EatBot stays within UberEats' core business and branding
- It enhances the current customer experience through its new features that still align with current UberEats
 - For example, by creating meal plans, EatBot transforms the limited filtering options that leaves everything up to the user, to creating detailed and tailored plans with little to no effort from the user
- Each new feature is a simple but effective way to leverage Uber's massive amounts of data to give users the impression that they are getting superior service on UberEats
- EatBot will allow UberEats to capture more market share and generate a competitive advantage over other delivery apps by demonstrating differentiation
- This will in turn allow for increased sales and profitability

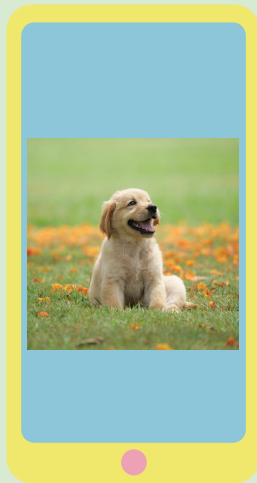


EXAMPLE USER INTERFACE



FOOD RECOGNITION: New Scenario

- Use general-image-recognition model to detect food-related elements
- If there isn't any food-related words, return the key elements revealed in the image
- Use food-item-recognition model to identify the Dish if certain keywords are detected
- Match the identified dish name to the restaurant dataset to find restaurants that have this dish
- Return the top three possible dish names, prices, and the relevant restaurants



```
The dish might be: steak
Restaurants that serve this dish:
- Bella Luna (Dish: T-Bone Steak, Price: 22.5)
-----
The dish might be: filet mignon
Restaurants that serve this dish:
- Bella Vita (Dish: Filet Mignon, Price: 18.0)
- Dolce Vita (Dish: Filet Mignon, Price: 22.5)
- Dolce Vita (Dish: Filet Mignon, Price: 16.5)
- Dolce Vita (Dish: Filet Mignon, Price: 22.5)
- Dolce Vita (Dish: Filet Mignon, Price: 21.0)
- Dolce Vita (Dish: Filet Mignon, Price: 25.0)
-----
The dish might be: sirloin
There is currently no store that contains this dish
-----
```

The image you uploaded contains elements cute, dog, grass, puppy. No food detected.

```
('beef', 0.9905311465263367), ('steak', 0.9830650091171265), ('dinner', 0.9776675105094
```

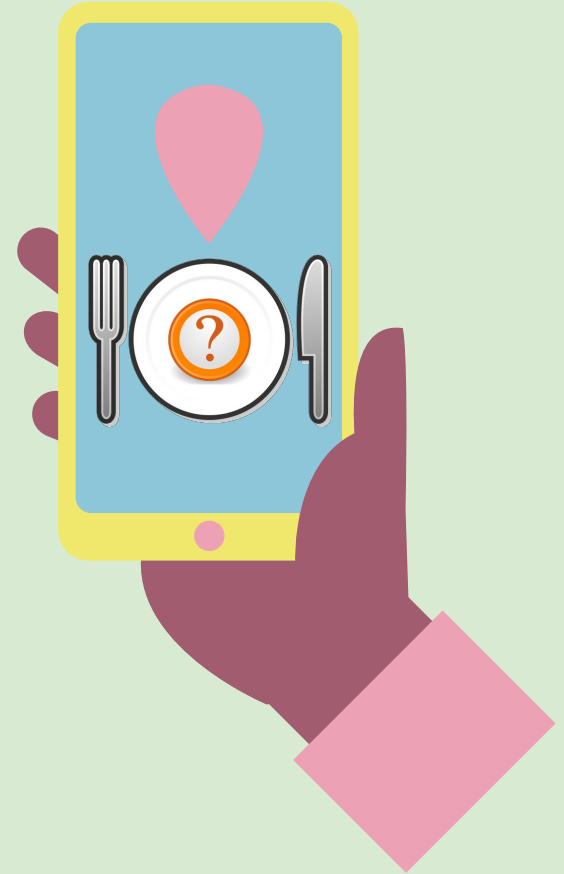
MEAL RECOMMENDATIONS & PLANNING: New Scenario



- Problems: Uber Eats has a relatively limited filter option; customers always get struggled on what to eat from too many choices
 - We use Chatbot to extend the filters to further narrow down customers' choices and make recommendations based on their inputs such as budget.
 - We also have the function of meal planning by ask customers to input how many meals they want to be recommended.
- Code: used Pandas library to read the mock data file and filter qualified restaurant based on customers' input for parameters
 - Parameters taken into consideration:
 - Budget, occasion, restaurant type (NOT want to eat for sure), maximum waiting time expected, minimum rating expected, and number of meals wanted to be recommended

SURPRISE FOOD: New Scenario

- Inspired by Too Good To Go app
- If the user wants to get surprised EatBot can surprise them by recommending a restaurant and its dish
- EatBot starts by asking user if they want to get surprised if they say yes a recommendation given
- If the user accepts it, then the dish added to the cart; otherwise another recommendation is given (see Appendix 2)
- This functionality is not included in UberEats yet
- Some customers struggle with the amount of choice available on Uber Eats
- It will help customers to try new dishes and increase revenues for the company



SURPRISE FOOD: New Scenario

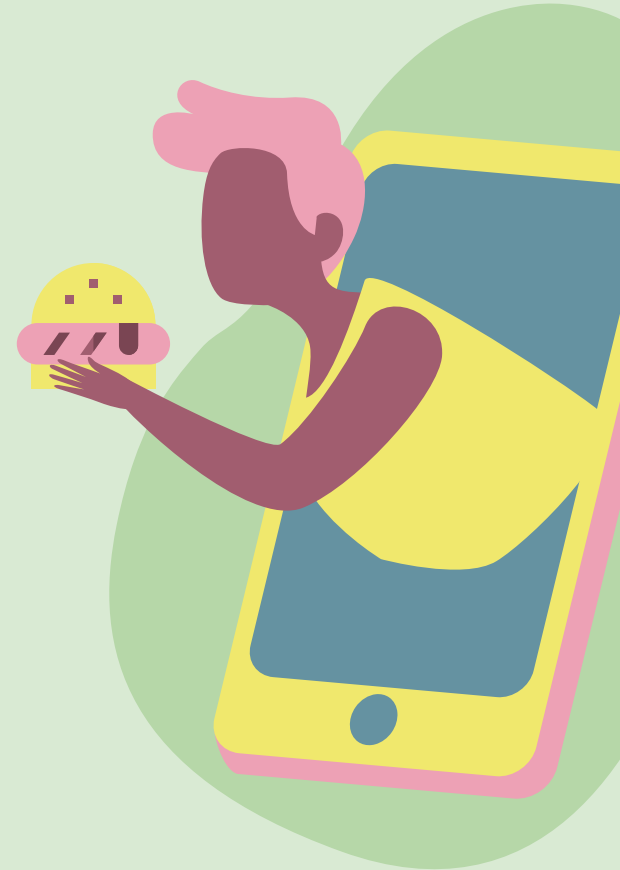
Python Libraries Used

1. 'random': Generates random values and selections for variety in the code's decision-making process (see Appendix 3)
2. 'CSV': Handles reading and writing CSV files, facilitating data processing from and to structured tables (see Appendix 4)
 - The 'random' library adds diversity to UberEats by enabling random dish and restaurant selections for surprise meals, while the 'csv' library facilitates organized storage and retrieval of restaurant data for efficient service.



EATS WRAPPED: New Scenario

- Currently, UberEats doesn't allow users to view insights on their data
- Inspired by Spotify's "Spotify Wrapped" and Apple Music's "Rewind"
- UberEats generates massive amounts of data that isn't leveraged to its full potential
- Allowing users to see their UberEats data in a fun and interesting way and be able to share it with others will:
 - Attract more consumers to the platform through word-of-mouth and posts online
 - Entice current users to interact more with the application over other delivery services

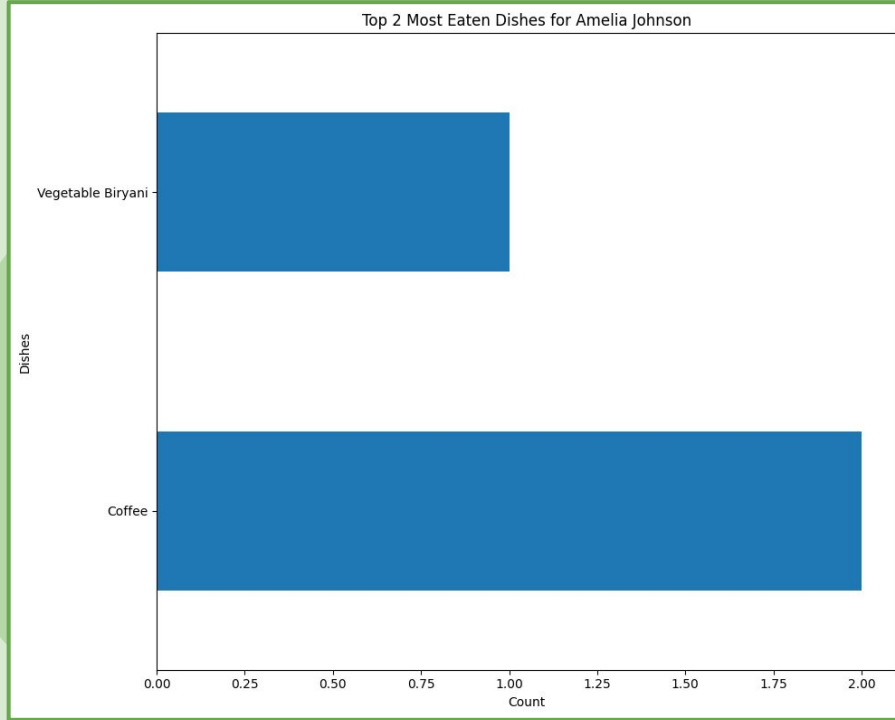


EXAMPLE 1

User's Top Dishes

Using matplotlib, EatBot provides a visualization of the user's current top dishes when prompted or when asked to generate a user's Eats Wrapped (see Appendix 5)

Theses visualizations, with the right graphic design and marketing team, can become fun infographics that users can share with friends or on social media



SHAREABLE INFOGRAPHIC EXAMPLES





Thank You!

Any Questions?

APPENDIX

- From our proposal, we considered four different scenarios including: UberEats Browsing, Meal Planning, Order Placement, and Customer Support. Since these topics are too broad and can contain millions of sub parts under each, we choose to select only one or two scenarios from our original brainstorm and expand them to be our new four scenarios for final submission.

Appendix 1: Meal Recommendation Sample Output

```
What is your name? a li s
Hi, how can I help you today?
Would you like to choose a service from below? (please enter a number)
    1. Searching for food
    2. I don't know what to eat today
    3. Surprise meal bag plz
    4. Check my EatsWrapped!

You: 2
Welcome to your meal recommendation! Let's start with some questions.
What's your maximum budget: 40
What's your occasion (please enter one word only): dinner
Is there anything you know you do not want to eat today (please enter one type of cuisine): american
What's your maximum expected time for waiting (please enter a number): 50
What's your minimum expected rating of the restaurant (out of 5): 3.5
How many meals you want for recommendations: 1
I would recommend [['Restaurant:Curry Delights', 'Dish: Mango Lassi$18.5']] for your 1 meals plan.
Are you happy with my recommendation (yes/no): no
Alright! Let's try another recommendation.
I would recommend [['Restaurant:Spice Delights', 'Dish: Naan$14.0']] for your 1 meals plan.
Are you happy with my recommendation (yes/no): no
Alright! Let's try another recommendation.
I would recommend [['Restaurant:Spicy Curry', 'Dish: Mango Lassi$15.0']] for your 1 meals plan.
Are you happy with my recommendation (yes/no): no
Alright! Let's try another recommendation.
I would recommend [['Restaurant:Ramen King', 'Dish: Takoyaki$10.5']] for your 1 meals plan.
Are you happy with my recommendation (yes/no): yes
Great, your meal is added to your cart!
```

Appendix 2: Surprise Food Sample Conversation

Chatbot: Hi, how can I help you today?

Chatbot: Would you like to choose a service from below? (please enter a number)

1. Searching for food
2. I don't know what to eat today
3. Surprise meal bag plz
4. Check my Eatswrapped!3

Are you interested in getting surprised today, yes/no?

yes

Great! I can surprise you with a restaurant's surprise meal today. Would u like me to consider your previous order history?

no

Great. I have selected Cannoli at Bella Vita for \$19.5. Are you happy with this selection?

no

I have selected Margherita Pizza at Napoli Ristorante for \$10.0 as an alternative. Are you happy with this selection?

sda

Invalid input. Please enter 'yes' or 'no'.

I have selected Margherita Pizza at Napoli Ristorante for \$10.0 as an alternative. Are you happy with this selection?

no

I have selected Risotto at Bella Trattoria for \$10.0 as an alternative. Are you happy with this selection?

yes

Great. The order has been added to your cart. Click on the checkout button to complete your purchase.

Appendix 3: Surprise Food Import Random & CSV

```
import random
import csv

def read_restaurant_data(filename):
    restaurant_dict = {}
    with open(filename, 'r') as file:
        csv_reader = csv.DictReader(file)
        for row in csv_reader:
            restaurant_name = row['Name']
            budget_range_dishes = eval(row['Budget Range Dishes'])
            restaurant_dict[restaurant_name] = budget_range_dishes
    return restaurant_dict

restaurant_data = read_restaurant_data('/content/drive/MyDrive/restaruant_data.csv')

def get_user_input(prompt):
    while True:
        user_input = input(prompt).lower()
        if user_input == "yes" or user_input == "no":
            return user_input
        else:
            print("Invalid input. Please enter 'yes' or 'no'.")

def select_random_dish(restaurant_data):
    selected_restaurant = random.choice(list(restaurant_data.keys()))
    selected_dish = random.choice(list(restaurant_data[selected_restaurant].keys()))
    selected_price = restaurant_data[selected_restaurant][selected_dish]
    return selected_dish, selected_restaurant, selected_price
```

Appendix 4: Surprise Food Function

```
def surprise_dish():
    conversation_active = True
    while conversation_active:
        q1 = get_user_input("Are you interested in getting surprised today, yes/no? \n").lower()
        if q1 == "yes":
            q2 = get_user_input("Great! I can surprise you with a restaurant's surprise meal today. Would u like me to consider your previous order history?\n")
            if q2 == "yes":
                selected_dish, selected_restaurant, selected_price = select_random_dish(restaurant_data)
                q3 = get_user_input(f"Great. I have selected {selected_dish} at {selected_restaurant} for ${selected_price}. Are you happy with this selection?\n")
                if q3 == "yes":
                    print("Great. The order has been added to your cart. Click on the checkout button to complete your purchase.")
                    conversation_active = False
                elif q3 == "no":
                    while True:
                        selected_dish, selected_restaurant, selected_price = select_random_dish(restaurant_data)
                        q4 = get_user_input(f"I have selected {selected_dish} at {selected_restaurant} for ${selected_price} as an alternative. Are you happy with this selection?\n")
                        if q4 == "yes":
                            print("Great. The order has been added to your cart. Click on the checkout button to complete your purchase.")
                            conversation_active = False
                            break
                        elif q4 == "no":
                            continue
                        else:
                            print("Invalid input. Please enter 'yes' or 'no'.")
            else:
                print("Invalid input. Please enter 'yes' or 'no'.")
        else:
            selected_dish, selected_restaurant, selected_price = select_random_dish(restaurant_data)
```

Appendix 5: EatsWrapped Chabot Sample Output

What is your name? Amelia Johnson
Hi, how can I help you today?
Would you like to choose a service from below? (please enter a number)

1. Searching for food
2. I don't know what to eat today
3. Surprise meal bag plz
4. Check my EatsWrapped! 4

Let's check your EatsWrapped data! I'm here to show you your food delivery stats.

You can ask me about your order history, your most eaten dishes, the number of dishes eaten, your app usage time, or the number of orders you've placed.

What would you like to see? Please enter one of the following:

Order history
Number of dishes
Eaten dishes
App usage time
Number of orders eaten dishes

Top 2 Most Eaten Dishes for Amelia Johnson

