Analysis Document

The UberEats Order Delivery/Pickup System

Course: ITEC 4010

GROUP MEMBERS

Onyinyechi Onyeagucha 216883456

Jordan Bayon 216424897

Ahmet Yavas 217274978

Analysis Strategy

As-Is System

Description

The current UberEats Order Delivery/Pickup System allows individuals to pick a specific restaurant from a wide variety of restaurants and place an order with the option of either having the order delivered or picking the order up from the restaurant. This system also has a payment system where customers can choose to pay with a credit card, debit card, or PayPa, but we are solely focused on the system that handles order delivery/pickup.

Problems

1. Inability to change order items after placing an order
2. Frequent order inaccuracy due to restaurant miscommunication
3. Insufficient safety measures
4. Inadequate customer support
5. Wait and delivery times are often too long
6. Inaccurate wait time estimates (depending on drivers available)

To-Be System

The new system will have the following functionalities added to the existing UberEats Order Delivery/Pickup system in order to solve the problems listed above:

1. Implementation of a window that allows the user to make changes to their order before it has been received by the restaurant. But once the restaurant gets the order, the user may only make changes to the order by making a direct phone call to the restaurant.
2. The app will provide a feature for the restaurant to use for the confirmation of order correctness.
3. The new system should use codes between the driver and the user to ensure the order only goes to the customer. It will also ensure that all its drivers have completed all safety training before they are hired.
4. UberEats customer support takes too long to respond to customer requests most of the time. So the new system will use a chatbot to respond to more accessible customer requests and a 24/7 response team to answer requests immediately.
5. In order to fix the wait and delivery time problems, the new system will use incentives to help motivate drivers. It will also only select drivers within a specific radius and work with the restaurants to provide optimized menu options for faster delivery choices.
6. The new system will incorporate information like traffic data and any possible order factors in order to improve the accuracy of the wait times.

Requirements Analysis

Questionnaire

1. Have you ever experienced any issues with changing your order items after making a payment on the UberEats app?
   Yes
   No

2. How satisfied are you with the current process of changing your order items after payment?
   Very satisfied
   Satisfied
   Neutral
   Dissatisfied
   Very dissatisfied

3. Have you ever experienced any order inaccuracies due to restaurant miscommunication while using UberEats?
   Yes
   No

4. How satisfied are you with the current process of addressing order inaccuracies due to restaurant miscommunication?
   Very satisfied
   Satisfied
   Neutral
   Dissatisfied
   Very dissatisfied

5. Do you feel that the safety measures for drivers and customers are adequate while using Uber Eats?
   Yes
   No
   If no, please specify which safety measures you believe need improvement:

6. Have you ever had any issues with customer support while using UberEats?
   Yes
   No
   If yes, please describe the issue(s) you experienced:

7. How satisfied are you with the current wait and delivery times on the UberEats app?

   Very satisfied
   Satisfied
   Neutral
   Dissatisfied
   Very dissatisfied

8. Have you ever experienced inaccurate wait time estimates on the UberEats app?
   Yes
   No

9. How satisfied are you with the current accuracy of wait time estimates on the UberEats app?
   Very satisfied
   Satisfied
   Neutral
   Dissatisfied
   Very dissatisfied

10. Is there any other feedback you would like to provide on your experience using UberEats?

    Thank you for taking the time to complete this questionnaire!

System Models

Process Modelling of the To-Be System: Data Flow Diagram

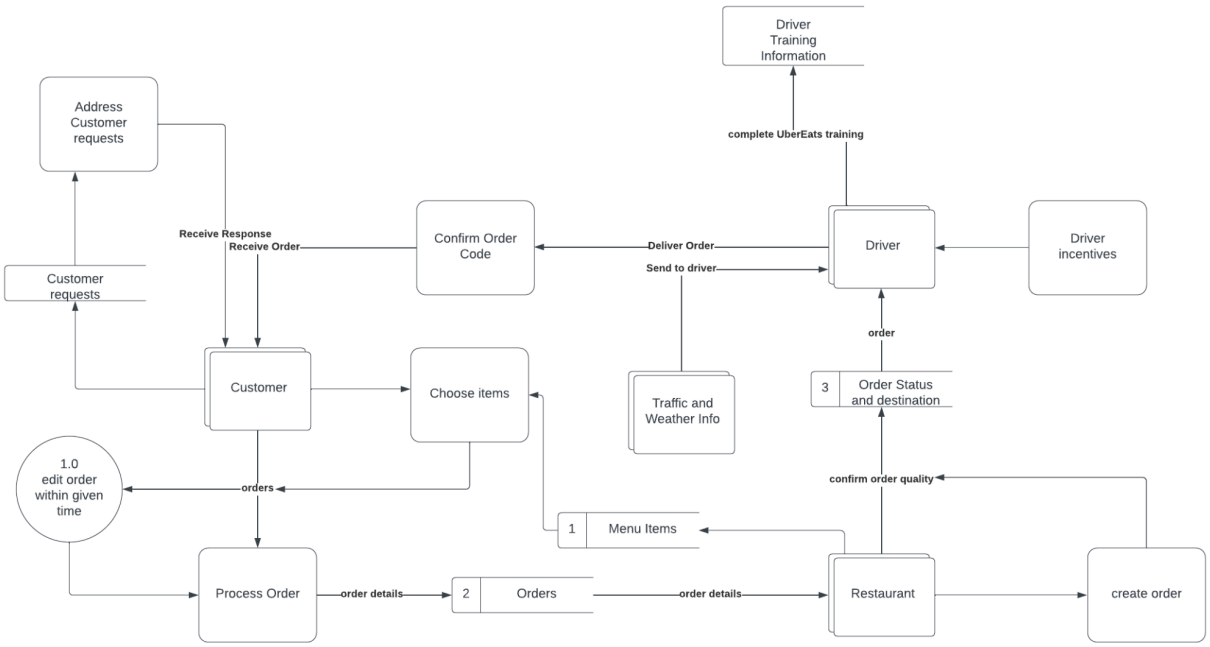


Diagram Description

In this diagram, the Customer is an external entity who chooses items from the Menu Options data store provided from the Restaurant. The order is processed and stored in the Orders data store. Before this, the Customer has a time period when they can change the details and options of their order, before it is processed. The external entity, Restaurant, receives the details stored in Orders so it can be created before its quality is confirmed. The details of the order's destination, alongside the confirmed quality of the order are stored in the Order Status and destination data store. The driver receives the information from the data store and begins the delivery process. In order to complete the delivery process, the driver is provided with information from external systems about the weather or current traffic and is required to confirm a code with the customer before they can finalize the order.

The customer's inquiries to Customer care and the chatbots are stored in the customer requests data store. Customer care representatives access the requests from the data store and send responses to these requests back to the Customer. The Drivers are required to complete training before they can begin making deliveries. The proof of training completion is stored in the Driver Training Information data store. Finally, In order to encourage the Drivers the uber eats management system initiates the process of providing it's drivers with various incentives. This keeps them motivated and more willing to put in their best effort at work.

Data Modelling of the To-Be System: Entity Relationship Diagram

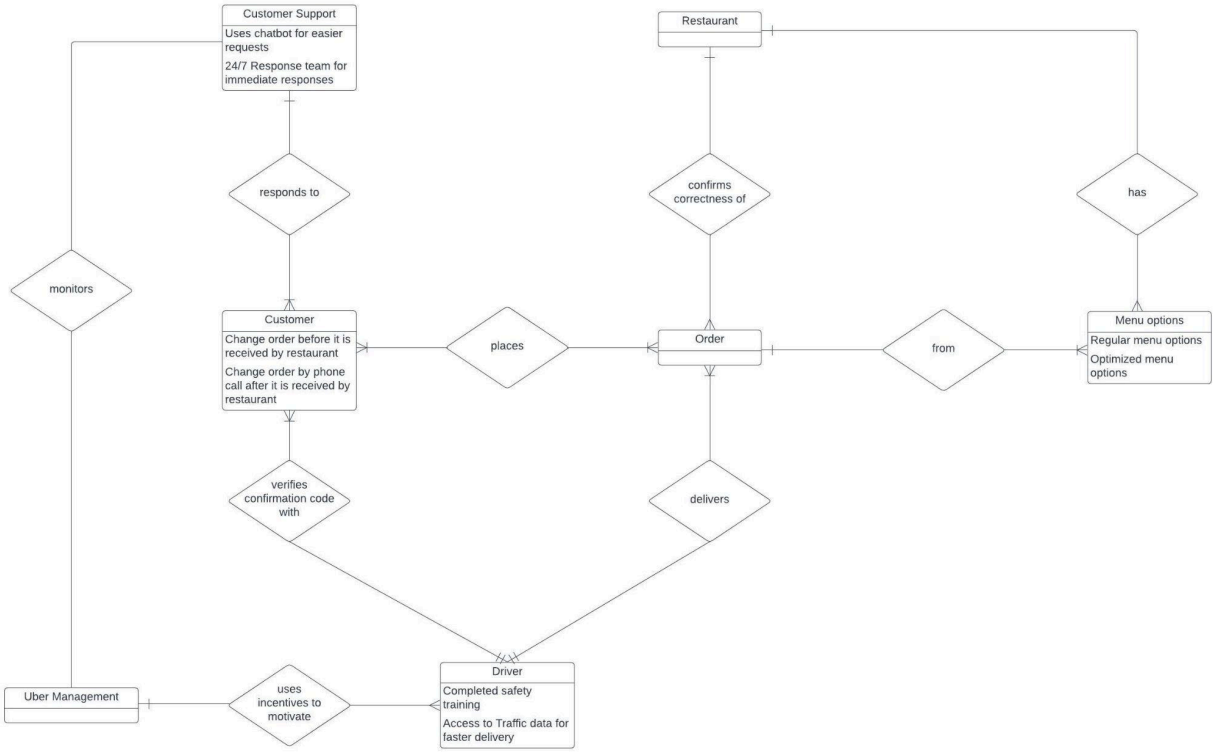


Diagram Description

In this diagram, the Customer places an Order with a relationship of one or many to one or many. Customers have two attributes, the ability to change orders before the restaurant has received them via the UberEats app and the ability to change the order after the restaurant has received it via a phone call. The Customer and the Driver have a code to verify orders, with a relationship of one and only one to one or many. Drivers have two attributes, proof of completion of safety training and access to traffic data to improve their delivery time. Restaurants confirm Order correctness before they are delivered. This has a relationship of one to many.

The Orders are delivered by Drivers with a relationship of one and only one to one or many. Orders are made from various Menu options, whose attributes are standard menu options and optimized menu options. Their relationship is one-to-one or many. The Menu options are provided by the Restaurants via a one-to-many relationship. Customer Support responds to User requests. They have two attributes, the ability to use a chatbot for more accessible responses and the ability to provide 24/7 support to the user. Customer support is monitored by the Uber management team. Finally, Drivers are provided with incentives by the Uber Management team to keep them working harder and faster. It has a one-to-many relationship.

Use Cases and Use Case Diagrams

Functional Requirements

- **The UberEats system requires four functional requirements that perform tasks:**
  - **Restaurant**
  - **Customer**
  - **Order**
  - **Delivery driver**
  - **UberEats chatbot**
  - **Customer service representative**

- **Restaurant:**
  - **Add menu items & restaurant information**
  - **Remove menu items & restaurant information**
  - **Update menu items & restaurant information**
  - **Confirm customer order**

- **Customer:**
  - **View restaurants**
  - **Search/filter restaurants by:**
    - **Distance**
    - **Price**
    - **Best rating**
    - **Most popular**
    - **Delivery time**
    - **Food type (japanese, indian, bbq, etc)**
  - **Make last minute changes** (apart of customer order use case)
  - **Order a delivery**
  - **Confirm order with driver** (apart of customer order use case)
  - **Contact UberEats chatbot**
- **Order:**
  - **Add item to order**
  - **Remove item from order**
  - **Leave order in cart**

- **Delivery driver:**
  - **Accept orders**
  - **Contact customer & restaurant**
  - **Confirm order with customer**
  - **Deliver food**
- **UberEats chatbot:**
  - **Respond to user requests**

    - **Assign 24/7 customer service member to request**
- **Customer support member:**
    - **Handles user requests**

Use Case Name: Create order request	**ID: UC-6**	**Priority: High**
Actor(s): Customer, Restaurant		
Description: The customer selects and orders food from a restaurant		
Trigger: Customer wants to order food		
Type: External		
Preconditions: **1. Customer is logged into their account** **2. Customer has a valid payment method** **3. Customer has a valid delivery address**		
Normal Course: **1.0 Customer orders food** **1. Customer picks restaurant** **2. UberEats shows restaurant menu** **3. Customer adds or removes item(s) from cart** **a. Items can be left in cart to stay and be ordered at another time, or discarded** **4. Preliminary order with cost estimate and tip options is created and displayed to customer** **5. Customer can return to step 3, confirm order, save for future consideration, or exit without saving** **a. Customer can choose to have ordered deliver to their desired location or** **b. Customer can choose to pick up order at restaurant** **6. Unconfirmed orders are left in customer cart** **7. Restaurant is notified of confirmed order requiring approval** **8. Restaurant confirms customer order** **9. Customer receives order code & delivery driver info**		
Postconditions: **1. Unconfirmed orders are left in customer cart** **2. Restaurant is notified of confirmed order requiring approval** **3. Restaurant confirms customer order**		

Use Case Name: Filter restaurant	ID: UC-6	Priority: High
Actor: Customer		
Description: Customer wants to filter restaurant by distance, delivery time, or by food type		
Trigger: Customer wants to order food		
Type: External		
Preconditions: **1. Customer is logged into their account**		
Normal Course: **1.0 Customer filters restaurants** **1. Customer clicks filter button next to search bar** **2. UberEats brings up all filter options** **3. Customer can sort restaurant by popularity, delivery time, best deals and overall, price range, max delivery fee, or by diet by selecting their filters and pressing apply** **4. UberEats brings up restaurants based on customers criteria**		
Postconditions:		

Use Case Name: Delivery driver delivers order	ID: UC-6	Priority: High
Actor: Delivery Driver		
Description: Delivery driver delivers customer's order from restaurant		
Trigger: Restaurant accepts customer order		
Type: External		
Preconditions: **1. Driver is logged into UberEats** **2. Driver presses the "Go" button to enable orders coming in** **3. Customer sends order to restaurant** **4. Restaurant accepts customer order**		
Normal Course: **1. UberEats notifies driver of available order to fulfill in their area** **2. Driver accepts order by pressing the "Tap to accept" button** **3. UberEats GPS shows driver order information including order code, and fastest rout to the location of the restaurant**		

4. **Driver drives to restaurant location**
5. **UberEats shows customer order information to driver**
6. **Driver asks restaurant for customer order**
    a. **If the order is not ready, the driver can report the issue by pressing the emergency button and selecting "Not ready" or "Report issue". If nothing is wrong, continue to step 7**
7. **Driver confirms food pickup by selecting "Start delivery" on UberEats**
8. **UberEats GPS shows driver the customer's location**
9. **Driver delivers order to customer's house**
    a. **Driver can contact customer by pressing the phone button on UberEats**
    b. **Customer may request to meet the driver in person or food to be left at the door**
10. **Driver completes order by swiping the "Delivered" button on the UberEats app**

Postconditions:

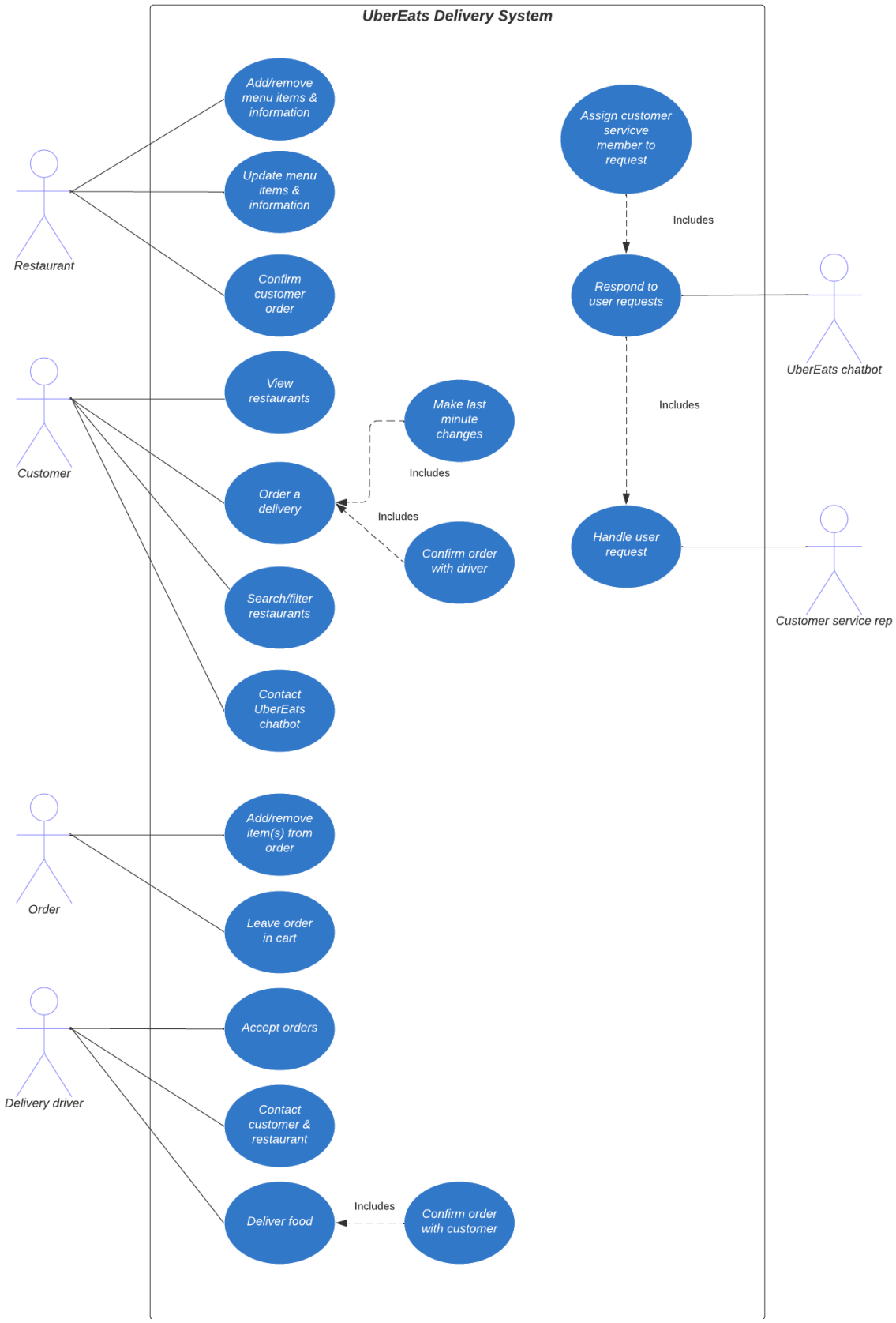
Use Case Name: Customer service request	**ID: UC-6**	**Priority: Medium**
Actor(s): Customer, UberEats chatbot, Customer service representative		
Description: Customer contacts the UberEats chatbot to request a customer service request with a customer service representative		
Trigger: Customer contacts UberEats chatbot		
Type: External		
Preconditions:		

Normal Course:

1. **Customer presses customer support button**
2. **UberEats chatbot greets customer, and brings up a list of possible customer issues**
3. **Customer selects one of the issues from the list, or enters a custom issue**
4. **UberEats chatbot redirects customer a customer service representative via online text chat**
5. **Customer service representative handles customer problem**
    a. **If issue can be resolved continue to step 6**
    b. **If issue is not resolved return to step 1**
6. **Customer receives compensation**

Postconditions:

1. **Customer receives compensation**


UberEats Delivery System
Add/remove menu items & information
Update menu items & information
Confirm customer order
View restaurants
Order a delivery
Search/filter restaurants
Contact UberEats chatbot
Make last minute changes
Confirm order with driver
Includes
Includes
Add/remove item(s) from order
Leave order in cart
Accept orders
Contact customer & restaurant
Deliver food
Confirm order with customer
Includes
Restaurant
Customer
Order
Delivery driver
Assign customer servicve member to request
Includes
Respond to user requests
Includes
Handle user request
UberEats chatbot
Customer service rep