Software Engineering

CS-4810

Fall 2023

Submitted by:

Omar Ali (202101259)

Ibrahim Mohamed (202000490)

Youssef Gamal (202100178)

Supervised by:

Dr. Aya Salama

Dr. Ola Galal

Table of Contents

1.1 Feasibility Study

Requirements Validation

- User requirement:
    - Menu:
        - Product picture and description:
            - Defined as Beverage or food.
            - Ingredients.
            - Customization/notes.
    - Cart:
        - Shows added items.
        - Count of each item.
        - Total count.
        - Add or Subtract from each item.
        - Delete item.
    - Checkout
        - Which card are you going to use.
        - Order for now.
        - Preorder with a time range.
        - Promo code section.
    - Profile:
        - Address is optional.
        - Name, age, number, sex/gender.
        - Payment information.
- Non-essential:
    - Reward system:
        - Get points for discounts or free products.
        - QR code that shows the cashier the number of points.

The requirements are valid and can be implemented and tested (except for IOS).

1.2 Economic Feasibility

The process of training and development will be done using free resources. However, the project might need payment for the usage of payment solutions/APIs.

1.3 Technical Feasibility

The application can run on web and mobile systems, however, testing can only be performed on android and web deployments because of resource limitations with how the Apple ecosystem works. For system integration with the restaurant/cafe, it is still in discussions with the client since the business isn't operating yet.

1.4 Operational Feasibility

For the usage of the application, only customers of the business will be using the app. For training of the employees, minimal training would be required for the branch managers and the employees working the register. Training would include learning how to interface with the admin side of the application as well as the processing of orders received from the application/users.

1.5 Schedule Feasibility

The time required to complete the main aspects/features of this project is predicted to be around 3 months. This includes the time required for the developers to learn Flutter. The main aspects/features of the project include the User Interface as well as the backend systems.

1.6 Project Community

Ibrahim Younes: Backend Developer

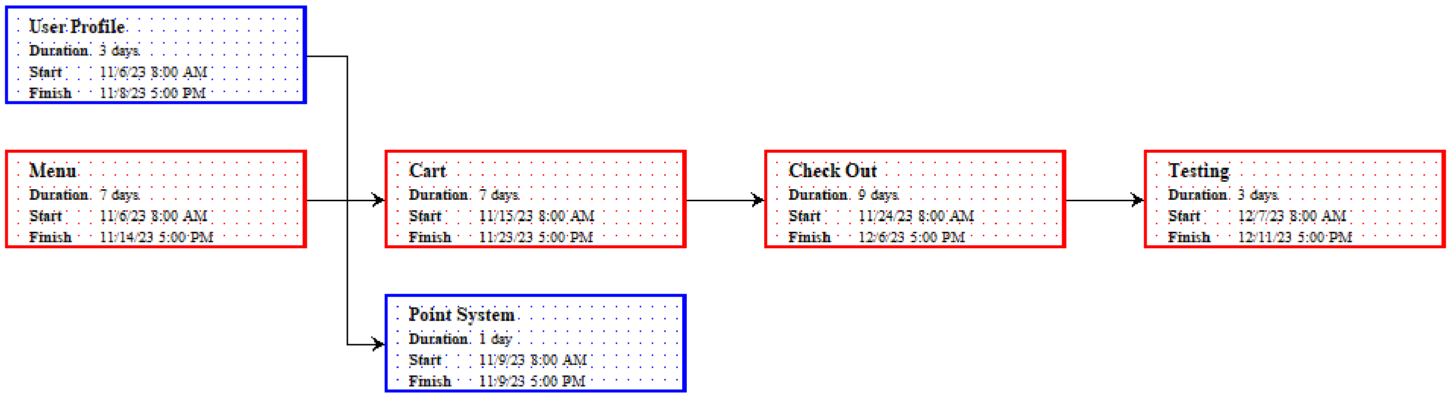
Youssef Hassan: Backend Developer

Omar Gaballah: UI/UX Designer/ Frontend Developer

1.7 Delivery Plan

	Milestone 1	Milestone 2	Milestone 3	Milestone 4	Milestone 5
Description	Feasibility Study Schedule Delivery plan	Priority of Tasks Dependency Chart Cost Estimate Key Requirments Task Analysis RMMM plan	Initial Class Definition Use Cases and Scenarios Use Case Diagram Activity Diagrams (for each use case)	State Diagram(s) Sequence Diagram(s) CRC Model	complete project with analysis and code
Assigned to	Everybody	Everybody	Everybody	Everybody	Everybody
Deadline	22/10/2023	5/11/2023	19/11/2023	3/12/2023	17/12/2023

1.8 Dependency Chart and Priority of Tasks:



1.9 Cost Estimate (SLOC):

Total effort = 3,000 LOC/ 1,000 LOC/person-month = 3 person-months
Total effort/3 = 1 months
Labor Rate = $20 per hour * 5 hours per person-day * 30 days per month = $3,000 per person-month
Total Cost = total effort * labor rate = 3 * $3,000 per person-month = $9,000

2.1 Technical Requirements

The application can run on web and mobile systems; however, testing can only be performed on Android and web deployments because of resource limitations with how the Apple ecosystem works. System integration with the restaurant/cafe is still being discussed with the client since the business isn't operating yet.

2.2 Operational Requirements

For the usage of the application, only customers of the business will be using the app. For training of the employees, minimal training would be required for the branch managers and the employees working on the register. Training would include learning how to interface with the admin side of the application as well as the processing of orders received from the application/users.

2.3 Tasks

Menu: food & beverage menu.

- Functionalities:
  - Product picture and description:
    - Defined as Beverage or food.
    - Ingredients.
    - Customization/notes.
- Priority: High.
- Effort: Medium.
- EST: 7 days.
- Acceptance Test: Has all the menu items, and the User can select them andadd them to the cart.

- Cart: A screen showing the items the user selected.
  - Functionalities:
    - Shows added items.
    - Count of each item.
    - Total count.
    - Add or Subtract from each item.
    - Delete item.
  - Priority: Medium.
  - Effort: Low.
  - EST: 7 days.
  - Acceptance Test: Shows the user their items and allows them to removesome or go for checkout.

- Checkout: A screen allowing the user to do a final check on their selected itemsand pay
  - Functionalities:
    - Which card are you going to use.
    - Order for now.
    - Preorder with a time range.
    - Promo code section.
  - Priority: Medium.
  - Effort: High.
  - EST: 9 days.

    - Acceptance Test: Checks out the user successfully and withdraws theamount.

- Profile: It has all the data of the user
    - Functionalities:
        - Address is optional.
        - Name, age, number, sex/gender.
        - Payment information.

    - Priority: High.
    - Effort: Low.
    - EST: 3 days.
    - Acceptance Test: Contains all the important data for each user.

- Testing: Tests the entire system.
    - Functionalities:
        - Check the entire program is working
    - Priority: High.
    - Effort: Medium.
    - EST: 3 days.

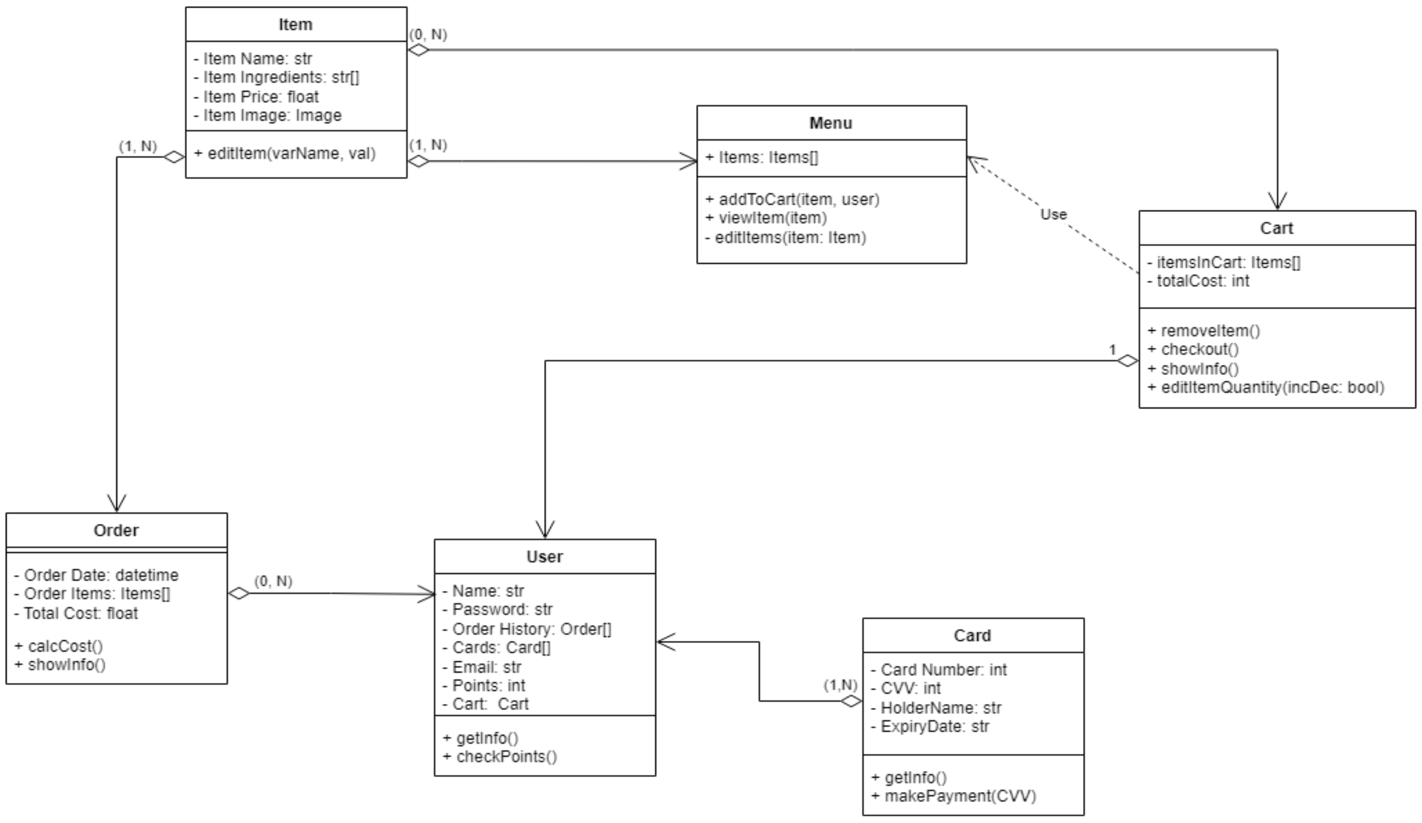
- Point system: Users get points with each purchase.
    - Functionalities:
        - Get points for a discount or free product.
        - QR code that shows the cashier the number of points.

    - Priority: Low.
    - Effort: Medium.
    - EST: 7 days.
    - Acceptance Test: The user can gain and spend points.

2.4 Risk Mitigation, Monitoring, and Management Plan

Risk ID	Name	Description	Category	Priority	Probability	Impact	Triggers	Potential Response
1	App Crashes	App stops working	Product	High	Medium	Serious	Too many inputs were given that the system couldn't handle. Unchecked errors	Better handling of user input and more focus on stress testing
2	Slow Loading	The app is slow to load.	product	medium	Low	Tolerable	Incompatible hardware with the app. the software is not optimized enough	Making sure that the app is as optimized and accessible as possible
3	Database Corruption	Database gets corrupted	product	Medium	Medium	Serious	Either due to system failure or database failure	Have backup data
4	Change of Requirments	The client changes the requirements	project	High	Medium	Catastrophic	The requirements weren't clearly defined with the client, which would cause a drastic change in the system	Dedicating more time to fully understanding the client. Some prototypes could be made to help
5	API failure	The APIs used in the checkout screen fail	product	High	Medium	Serious	Updates to the APIs used could break how the integration was made	With enough research, interfacing the APIs in a generic way should avoid the issue

2.5 Initial Class Definition:



2.6 Use Cases and Scenarios

Case 1:

Use Case Thumbnail: The user opens the app.

Use Case Description: This Use Case describes the process of the user when they open the app for the first time.

Preconditions: None

Postconditions: Users can view the menu.

Actors: User, Admin, and System

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- user opening the app.
- User Chooses between sign in and sign up. (A1)
- User Signs in.
- System checks credentials. (A2)
- Credentials are valid.
- User is logged in.
- User views the menu.

Alternative Flow:

<A1- If the user doesn't have an account, they choose to sign up.>

<A2- if the credentials are invalid, then an error message will be displayed to the user, prompting them to enter the correct credentials.>

Exceptions: None.

Constraints: None.

User Interface specifications: Login page, sign-up page, sign-in page.

<u>**Case 2:**</u>

Use Case Thumbnail: The user views the menu and adds items to the cart.

Use Case Description: This Use case describes the process of the user viewing the menu and choosing items to add to the cart.

Preconditions: The user logged in successfully.

Postconditions: items added to the cart successfully.

Actors: User, Admin, and System

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- user viewing the menu.
- User Chooses desired items and add-ons to add to the cart.
- System checks if any of the items are out of stock. (A1)
- Items are in stock.
- Items added successfully.

Alternative Flow:

<A1- If any of the items are out of stock, then an out-of-stock message will appear next to the items and will prompt the user to choose something different.>

Exceptions: None.

Constraints: None.

User Interface specifications: Menu.

Case 3:

Use Case Thumbnail: The User views and edits their cart.

Use Case Description: This Use case describes the process of the user viewing the cart.

Preconditions: The user logged in and added items successfully.

Postconditions: The user can edit items in the cart and can go to checkout successfully.

Actors: User, Admin, and System.

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- User views the cart.
- User edits selected items in the cart.
- System checks if edited items are in stock. (A1)
- Items are in stock, and the order is edited successfully.

Alternative Flow:

<A1- If any of the items are out of stock, then an out-of-stock message will appear next to the items and will prompt the user to choose something different.>

Exceptions: None.

Constraints: None.

User Interface specifications: Cart.

Case 4:

Use Case Thumbnail: The user goes to check out.

Use Case Description: This Use case describes the process of the user going to the check-out section to confirm the order and pay.

Preconditions: The user logged in,added items to the cart, and viewed the cart successfully.

Postconditions: payment is processed, and the order goes through.

Actors: User, Admin, and System.

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- User goes to check out.
- Systems check if points are enough for any discounts.
- User chooses a payment method (Credit or Cash).

Credit:

- User chooses a credit card (A1).
- User confirms order for payment.
- System Checks if the balance is enough. (A2)
- Payment proceeds successfully.
- Order is sent to the Café.

Cash:

- User confirms Order for Payment.
- The user is given a QR code with the order.
- User shows the QR code to the cashier.
- The cashier scans the code.
- User pays.
- Order starts.

Alternative Flow:

<A1- If the user doesn't have a credit card added to their profile, then they are asked to add one.>

<A2- If the balance is insufficient for the transaction, the order doesn't go through and prompts the user to choose a different payment method.>

Exceptions: None.

Constraints: None.

User Interface specifications: Check out the screen.

Case 5:

Use Case Thumbnail: The User Cancels the Order.

Use Case Description: This Use case describes the process of the User canceling the order.

Preconditions: User already ordered and can view order history from profile.

Postconditions: Order canceled successfully.

Actors: User, Admin, and System

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- User Cancels the order.
- The order isn't done yet. (A1)
- User gets a refund.

Alternative Flow:

<A1- If the order is already done. If the user is at the café, they receive the order. If the user is not at the café, then they are reimbursed in points >

Exceptions: None.

Constraints: None.

User Interface specifications: Order History Screen.

Case 6:

Use Case Thumbnail: The User views order history.

Use Case Description: This Use case describes the process of the User viewing and reordering an old order.

Preconditions: Users have an account and ordered before.

Postconditions: Users can reorder the same order again.

Actors: User, Admin, and System

Use Case Relationships: Associated with actors: User, System.

Basic Flow (Text):

- User views order history.
- User reorders an old order (A1)
- Order items are added to the cart.

Alternative Flow:

<A1- if User doesn't reorder, then nothing happens.>

Exceptions: None.

Constraints: None.

User Interface specifications: Order History Screen.

Case 7:

Use Case Thumbnail: The admin edits the menu.

Use Case Description: This Use case describes when the admin is editing the menu.

Preconditions: None

Postconditions: Menu is updated successfully.

Actors: User, Admin, and System

Use Case Relationships: Associated with actors: Admin, System.

Basic Flow (Text):

- Admin Chooses to add, remove, or update items.

- System checks if it’s working hours (A1).
- Not working hours, then the System applies the updated menu.

Alternative Flow:

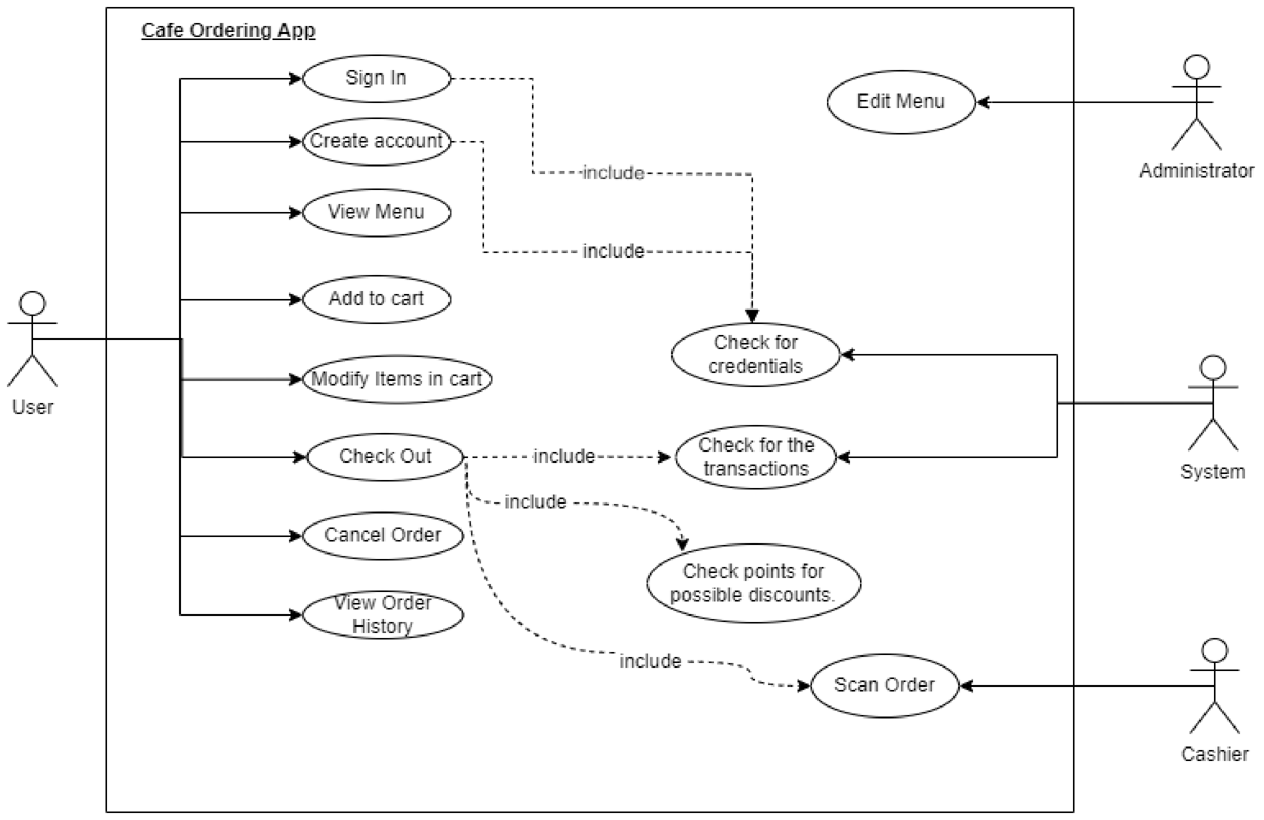
<A1- If it’s working hours, the system waits till it’s not working hours to apply the updates.>

Exceptions: None.

Constraints: None.

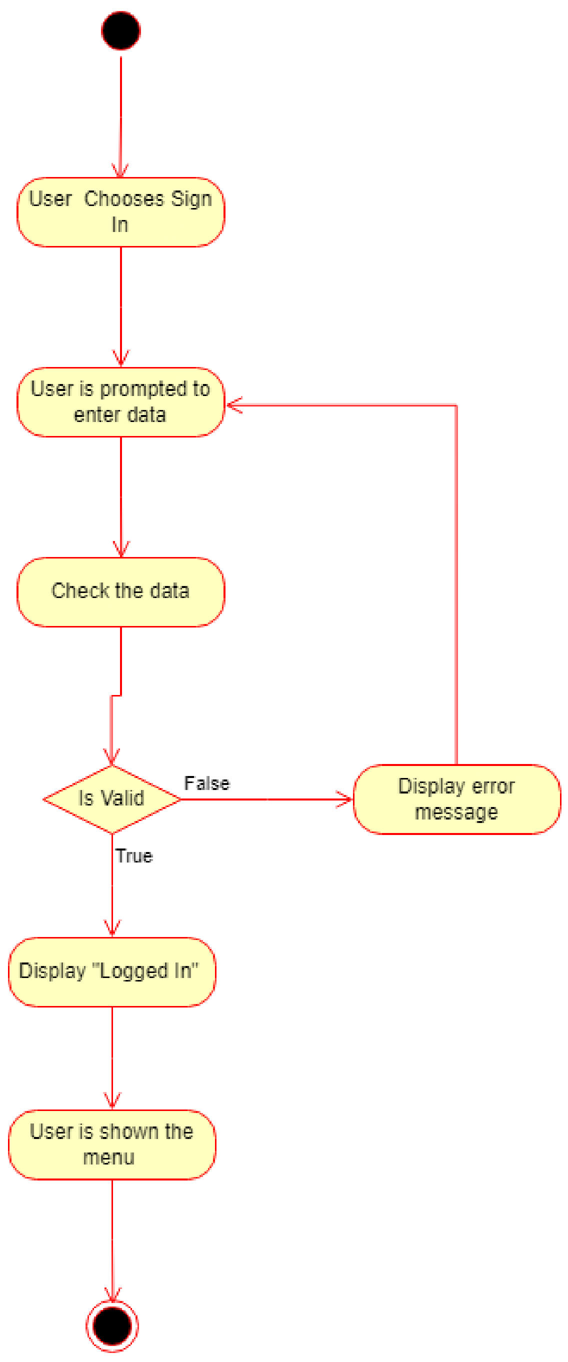
User Interface specifications: Admin menu screen.

2.7 Use Case Diagram

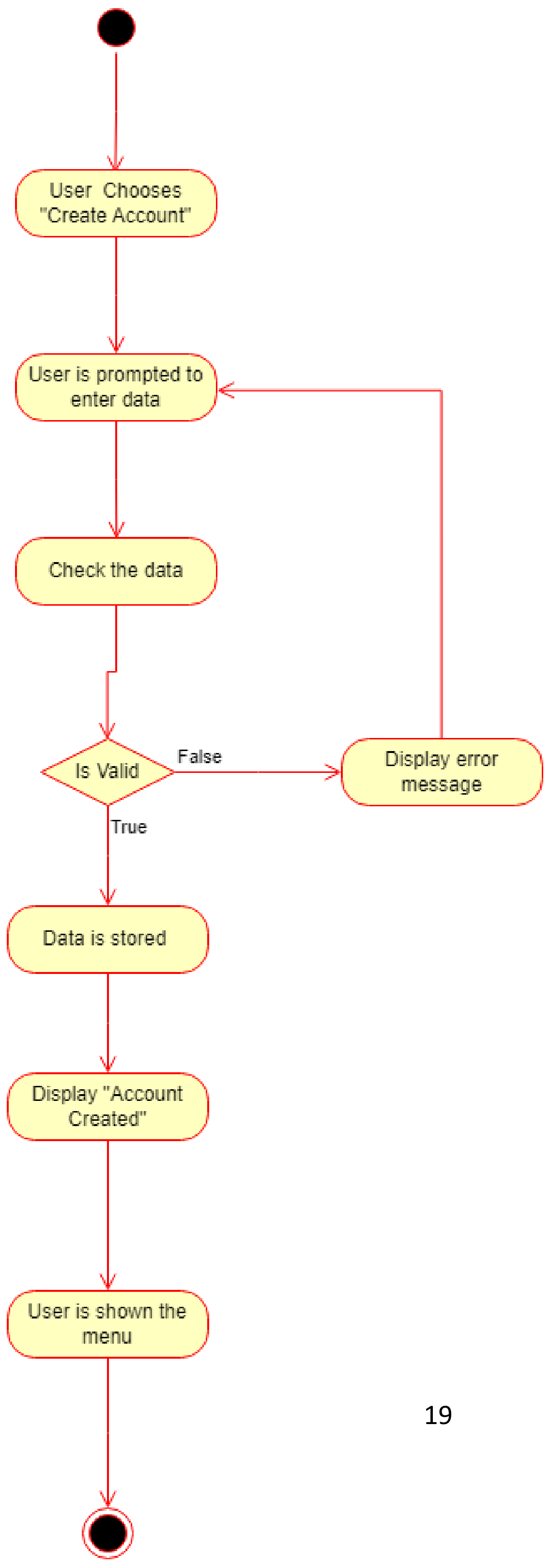


2.8 Activity Diagrams

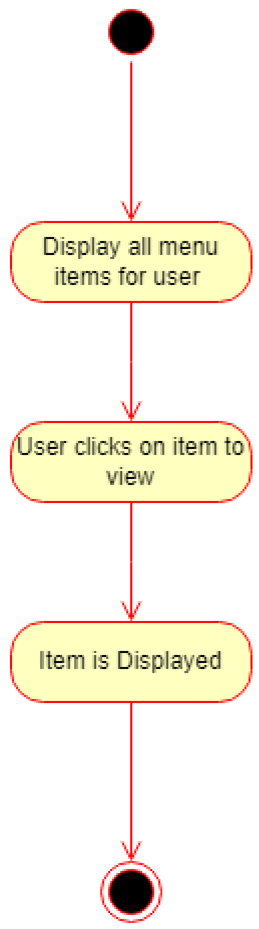
SIGN IN



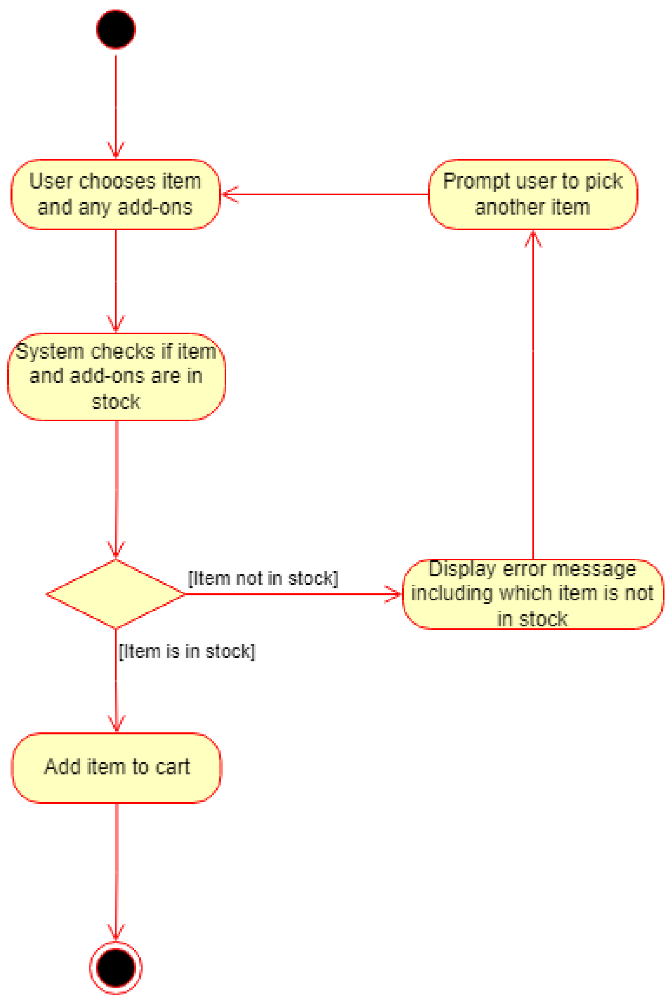
Create Account



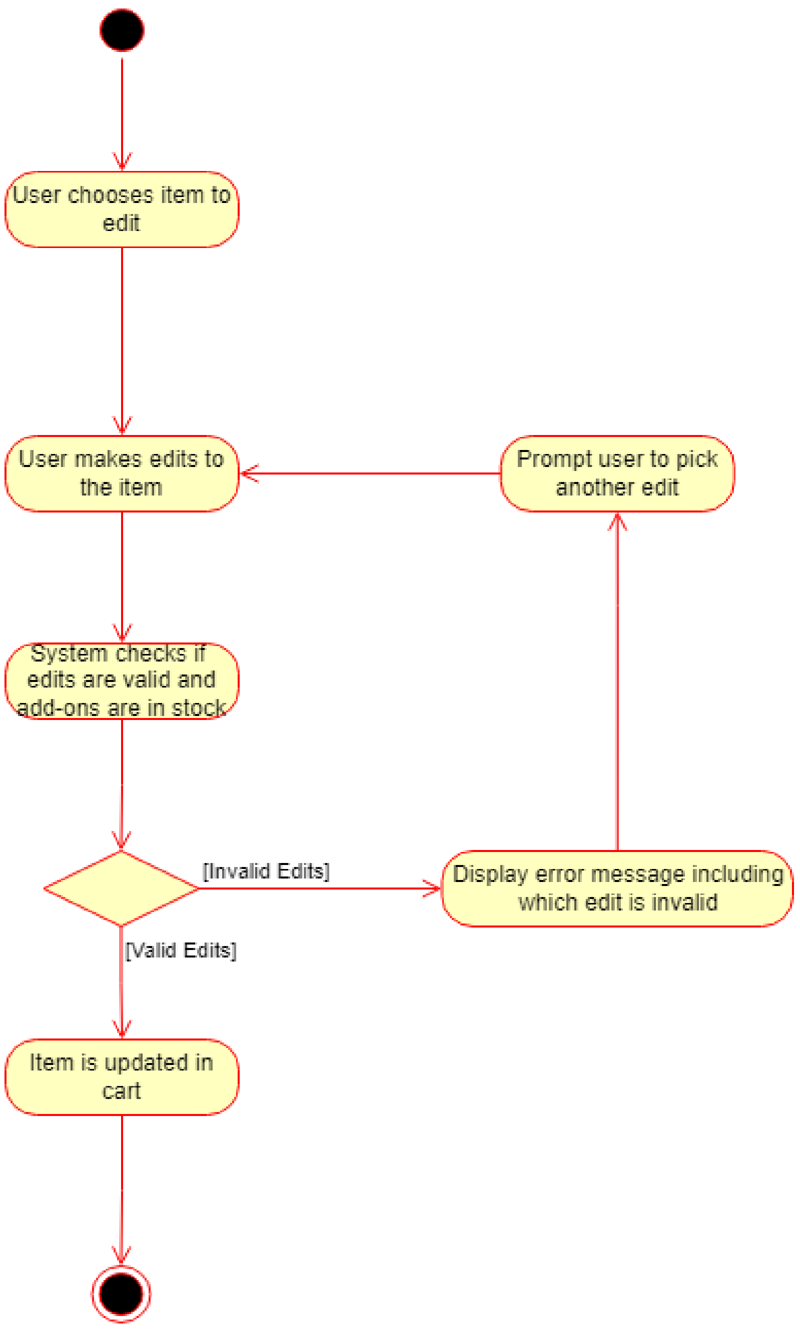
View Menu



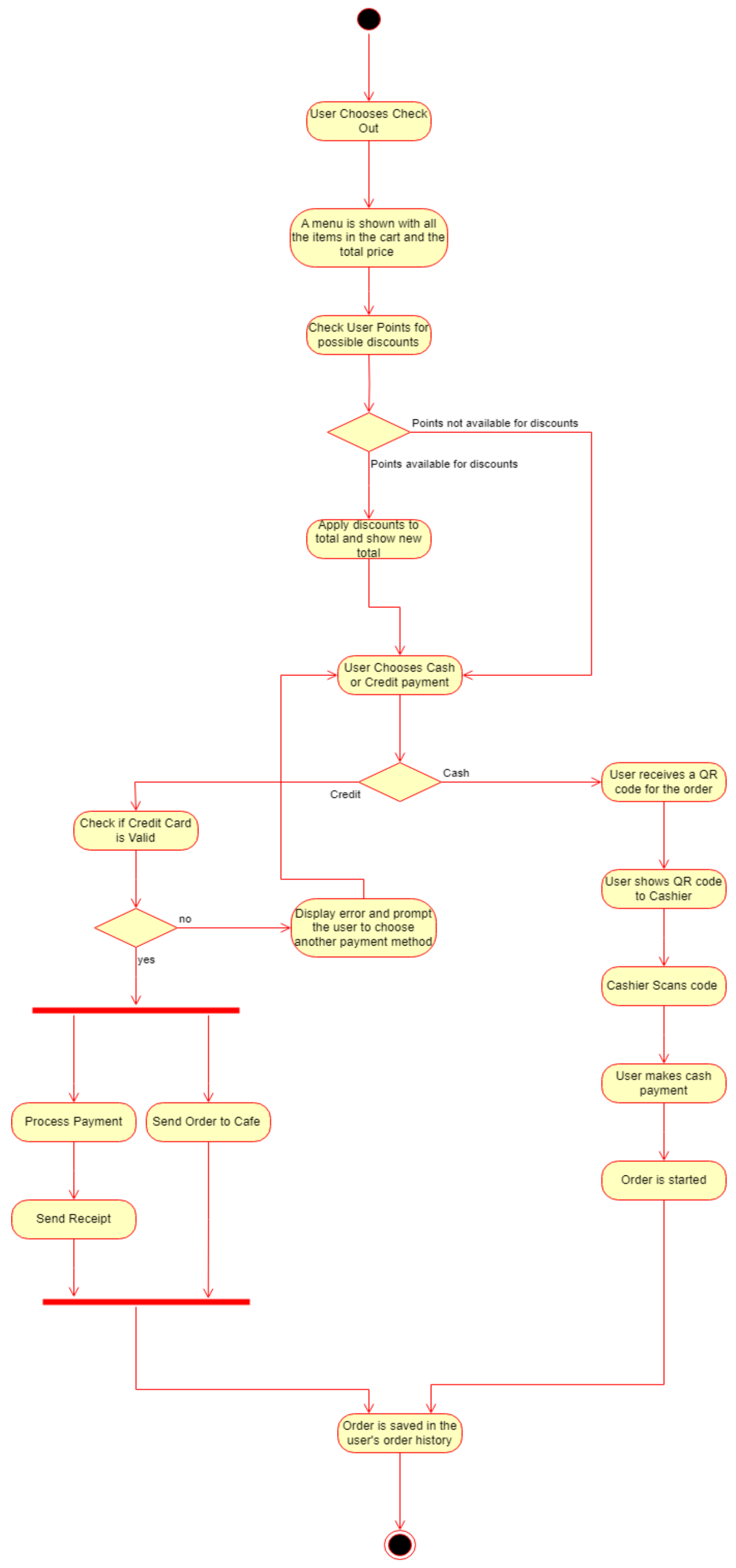
Add To Cart


User chooses item and any add-ons
Prompt user to pick another item
System checks if item and add-ons are in stock
[Item not in stock]
Display error message including which item is not in stock
[Item is in stock]
Add item to cart

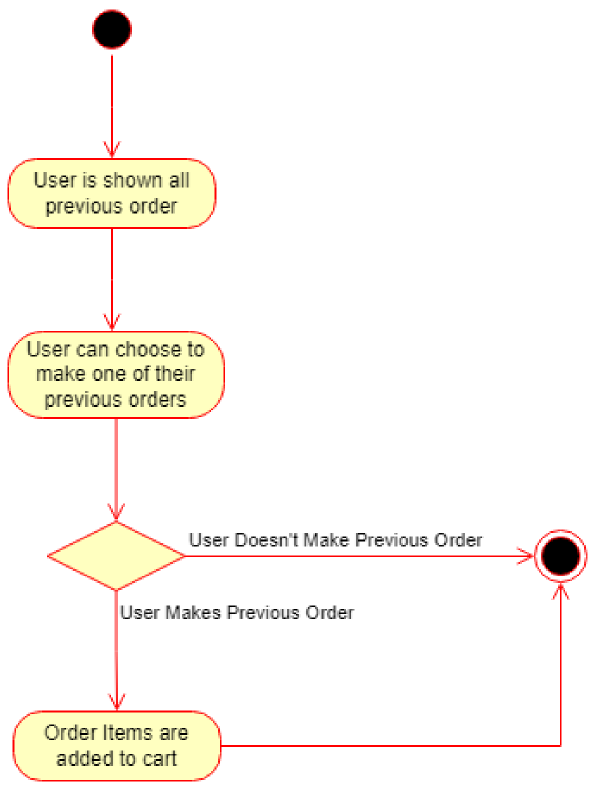
Modify Items in Cart


User chooses item to edit
User makes edits to the item
Prompt user to pick another edit
System checks if edits are valid and add-ons are in stock
[Invalid Edits]
Display error message including which edit is invalid
[Valid Edits]
Item is updated in cart

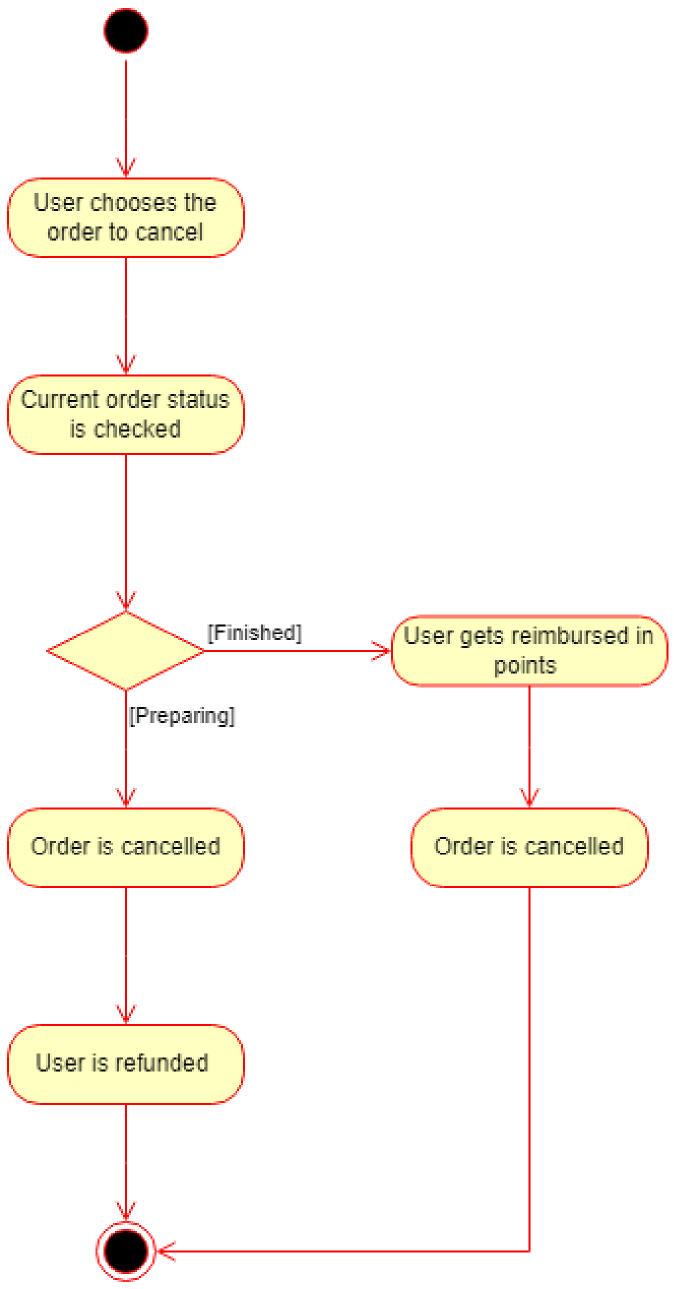
Check Out


User Chooses Check Out
A menu is shown with all the items in the cart and the total price
Check User Points for possible discounts
Points not available for discounts
Points available for discounts
Apply discounts to total and show new total
User Chooses Cash or Credit payment
Cash
Credit
User receives a QR code for the order
Check if Credit Card is Valid
User shows QR code to Cashier
no
Display error and prompt the user to choose another payment method
yes
Cashier Scans code
User makes cash payment
Process Payment
Send Order to Cafe
Order is started
Send Receipt
Order is saved in the user's order history

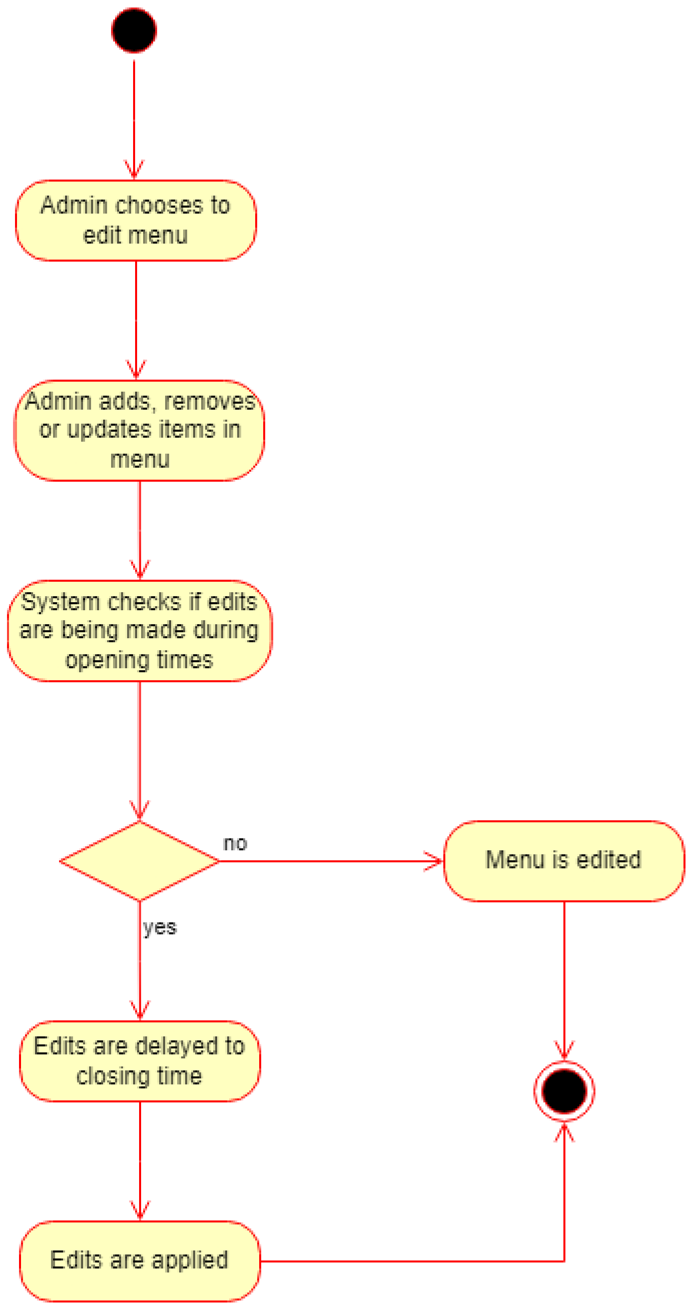
View Order History


User is shown all previous order
User can choose to make one of their previous orders
User Doesn't Make Previous Order
User Makes Previous Order
Order Items are added to cart

Cancel Order

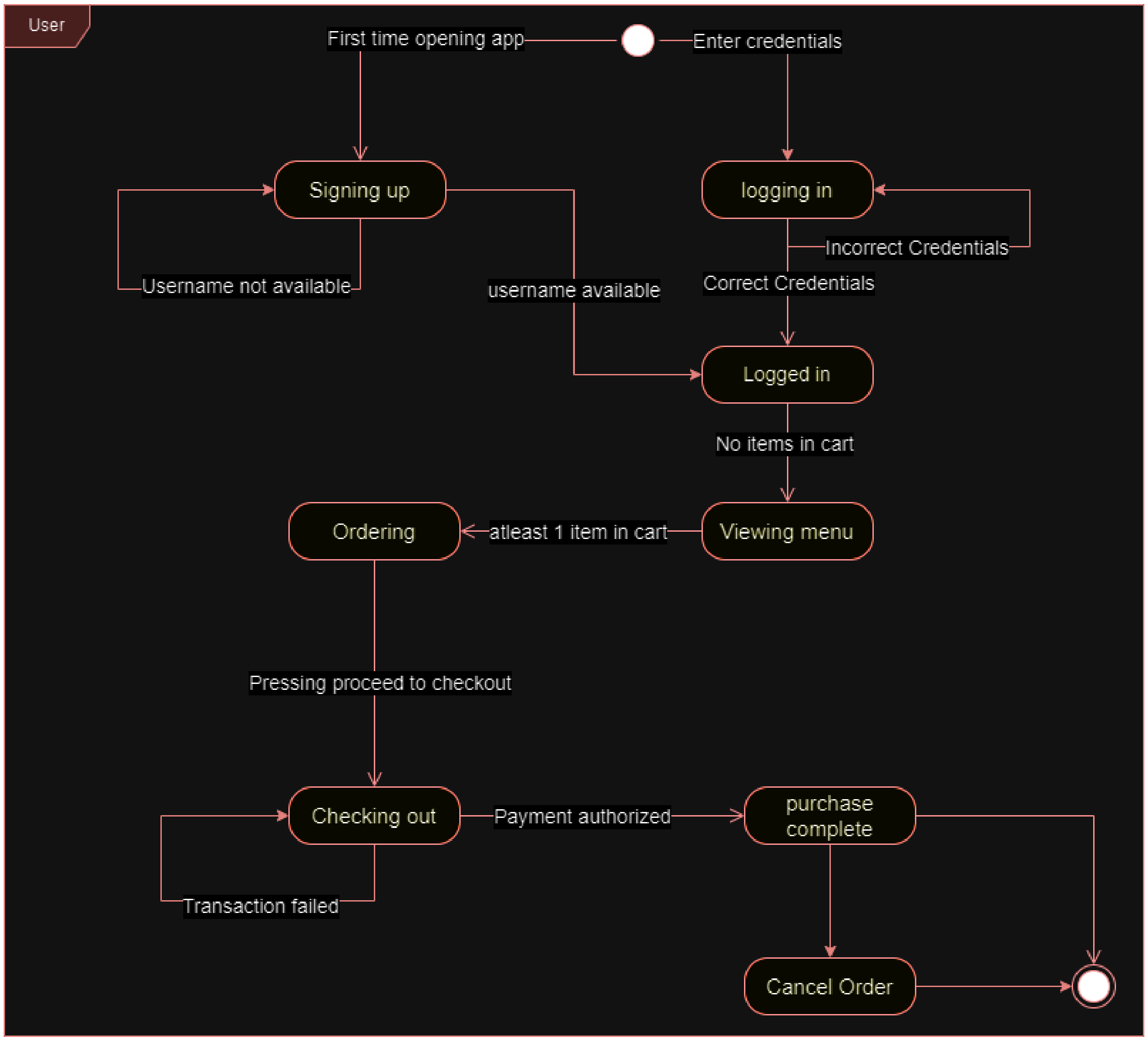

User chooses the order to cancel
Current order status is checked
[Finished]
User gets reimbursed in points
[Preparing]
Order is cancelled
Order is cancelled
User is refunded

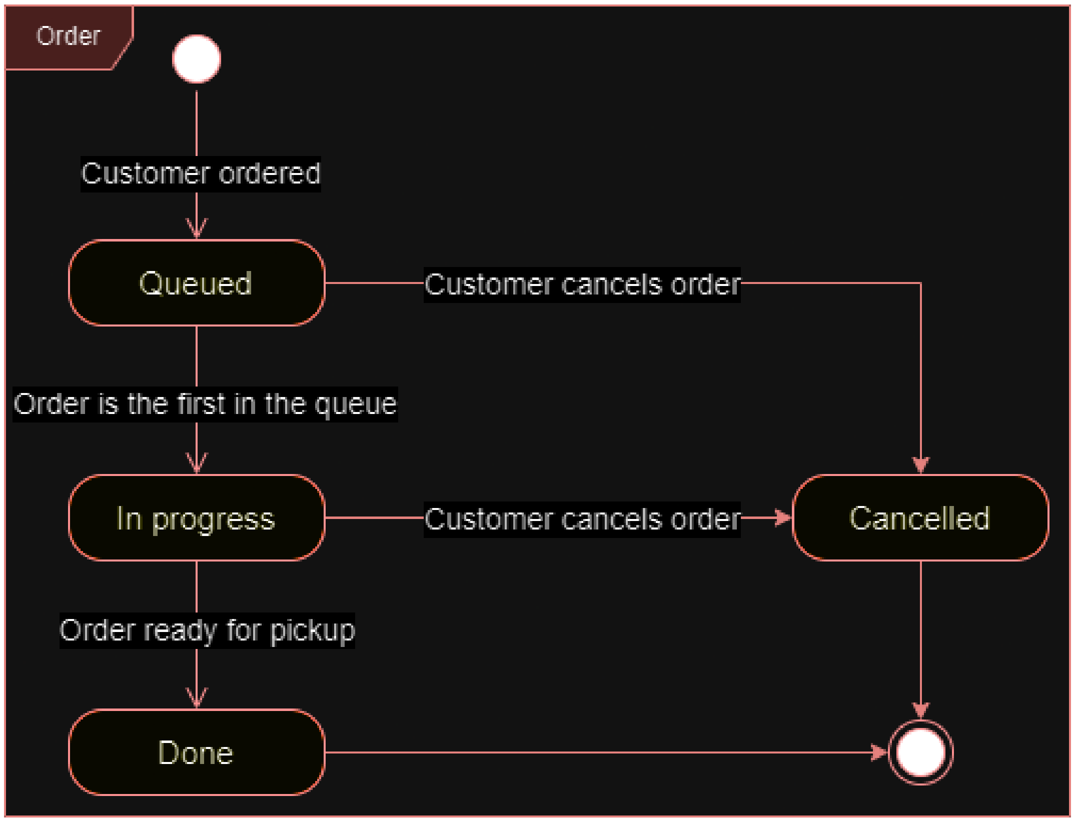
Edit Menu

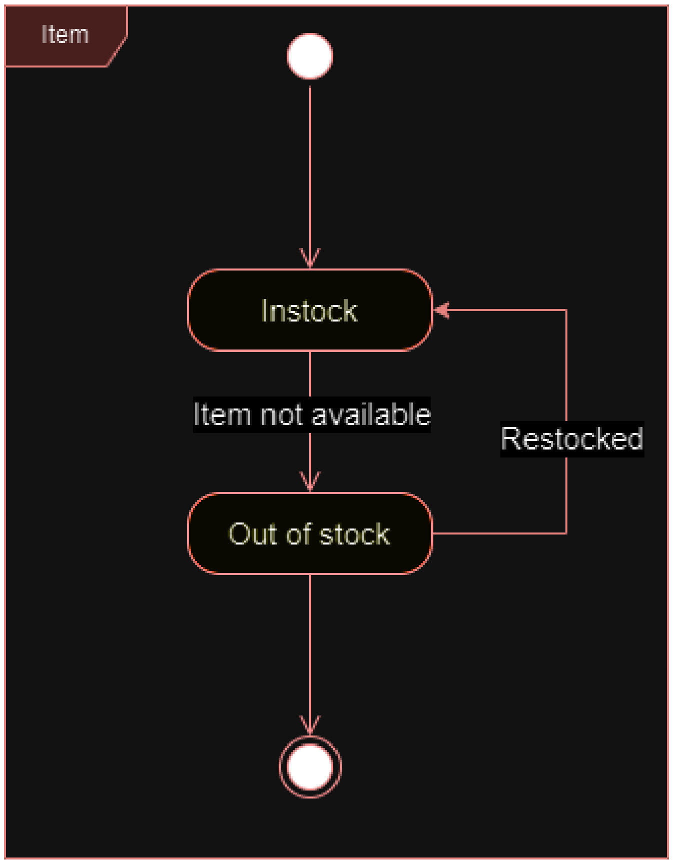


Admin chooses to edit menu
Admin adds, removes or updates items in menu
System checks if edits are being made during opening times
no
Menu is edited
yes
Edits are delayed to closing time
Edits are applied

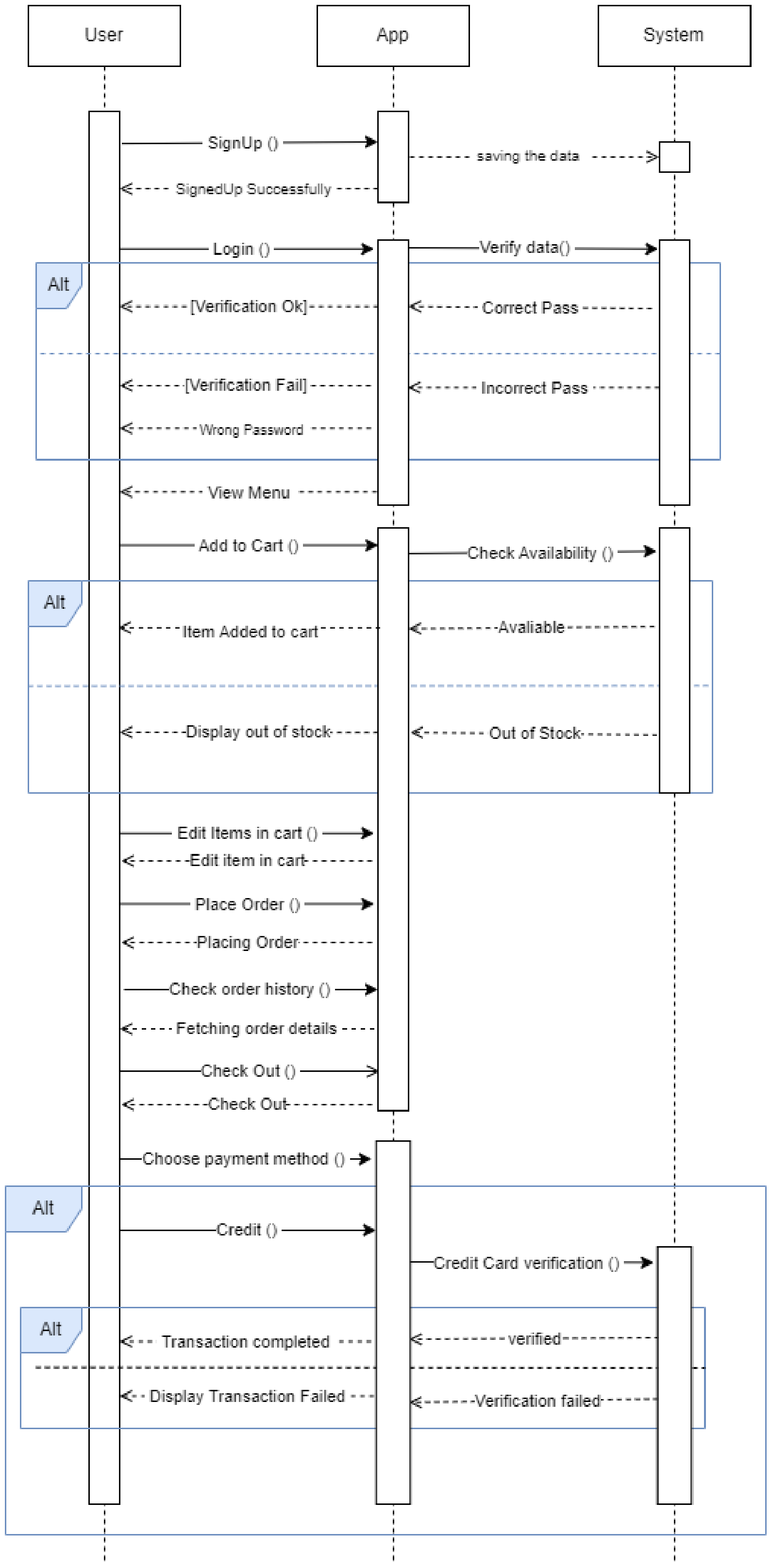
2.9 State Diagrams



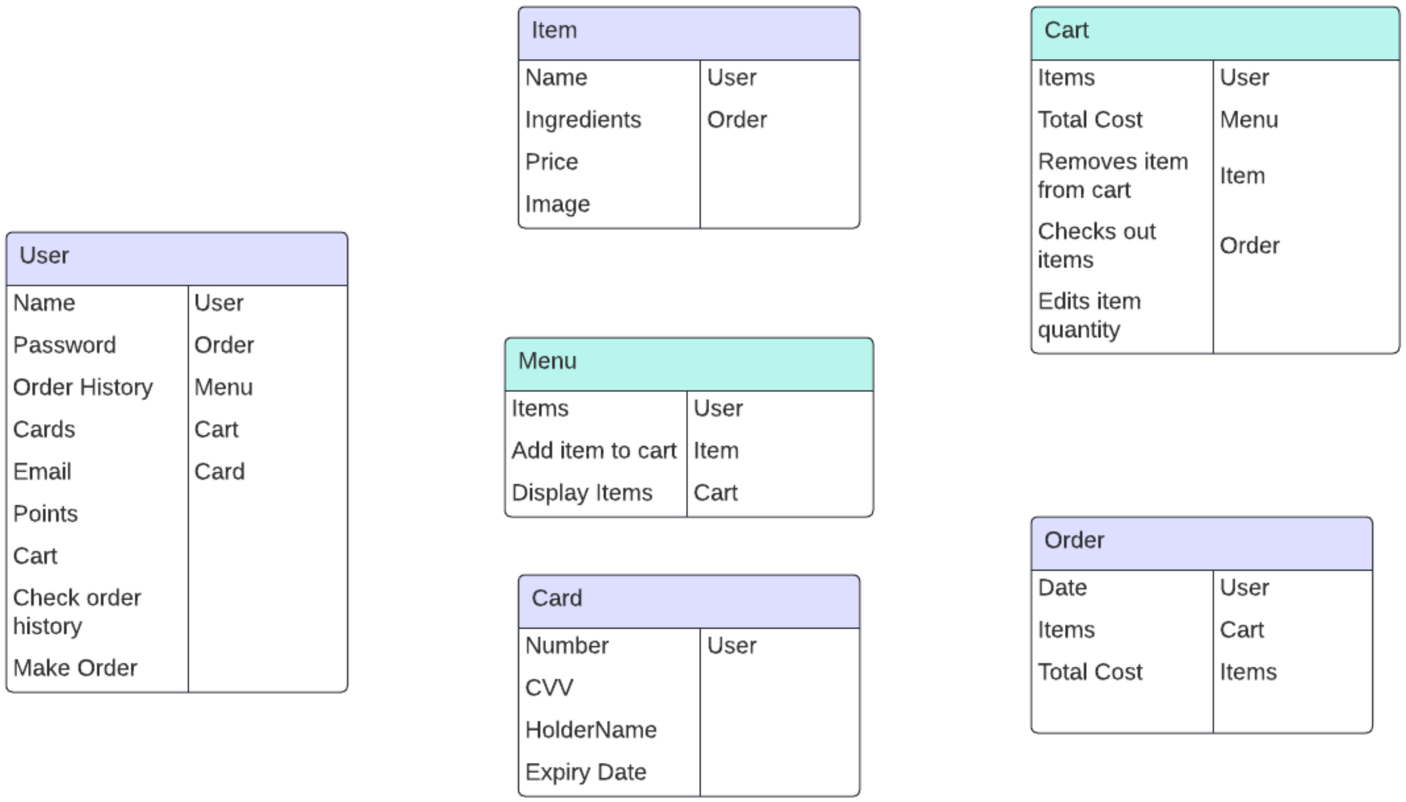

Order
Customer ordered
Queued
Customer cancels order
Order is the first in the queue
In progress
Customer cancels order
Cancelled
Order ready for pickup
Done


Item
Instock
Item not available
Restocked
Out of stock

3.1 Sequence Diagram



3.2 CRC Model



Item	
Name	User
Ingredients	Order
Price	
Image	

Cart	
Items	User
Total Cost	Menu
Removes item from cart	Item
Checks out items	Order
Edits item quantity	

User	
Name	User
Password	Order
Order History	Menu
Cards	Cart
Email	Card
Points	
Cart	
Check order history	
Make Order	

Menu	
Items	User
Add item to cart	Item
Display Items	Cart

Card	
Number	User
CVV	
HolderName	
Expiry Date	

Order	
Date	User
Items	Cart
Total Cost	Items

3.3 Project Repo:

https://github.com/ibrahim-younes/Softwere_Engineering_Project.git