

Course Title: Software Engineering (Sessional)

Course Code: CSE-434



CUET Food Delivery Service(Mobile App)

Team Members

Md. Al - Amin Bhuiyan
ID: 1804084

Fahimul Alam Araf
ID: 1804089

Mirza Asif Razoan
ID: 1804093

Department of Computer Science and Engineering (CSE)
Chittagong University of Engineering & Technology (CUET)
Chattogram- 4349, Bangladesh

Team Members

Photo	Name	ID	Email	Contact NO.	Total Credit Passed	CGPA acquired
	Md. Al - Amin Bhuiyan	1804084	amialaminb@gmail.com	01517814658	140.75	3.55
	Fahimul Alam Araf	1804089	araf04021@gmail.com	01614-246654	140.75	3.30
	Mirza Asif Razoan	1804093	mirzaasifrazoan@gmail.com	01755-233297	140.75	3.69

Name	ID	Contribution
Md. Al-Amin Bhuiyan	1804084	35%
Fahimul Alam Araf	1804089	30%
Mirza Asif Razoan	1804093	35%

Table 1: Contribution Table

Executive Summary

The CUET Food Delivery Service project is a meticulously planned endeavor, encompassing a comprehensive Software Requirements Specification (SRS) documentation for each module. These documents are crafted with precision, following industry-standard formats and benefiting from insights provided by SoftRobotics professionals.

Within the SRS, statements for three pivotal modules have been succinctly articulated. These statements have undergone necessary modifications to remain aligned with contemporary tools and technologies, ensuring the project stays at the forefront of industry trends.

Furthermore, the project's analysis models have been rejuvenated to provide a thorough understanding of the system's intricacies. This includes the integration of Use Cases, Sequence and Swim-lane diagrams, Activity and State diagrams, as well as Class and Class Collaboration diagrams. These models collectively foster a holistic comprehension of the project's architecture and functionality.

To enhance clarity and streamline development, a unified yet distinct Use Case has been devised. This encapsulates all three modules, emphasizing their interconnectivity and functional autonomy within the system.

Overall, the project strategically leverages a blend of robust documentation and insightful analysis models. This approach lays a strong foundation for the development of an efficient and user-centric CUET Food Delivery Service, ensuring it meets the diverse needs of its user base while remaining adaptable to future requirements and technological advancements.

Contents

List of Figures	7
------------------------	----------

1 Software Requirements Specification (SRS) for User Management in CUET Food Delivery Service	10
1.1 Introduction	10
1.1.1 Purpose	10
1.1.2 Scope	10
1.2 Overall Description	10
1.2.1 Product Perspective	10
1.2.2 User Classes and Characteristics	10
1.3 Functional Requirements	10
1.3.1 User Registration	10
1.3.2 User Authentication	11
1.3.3 Profile Management	11
1.3.4 Order History	11
1.4 Non-functional Requirements	11
1.4.1 Performance	11
1.4.2 Security	11
1.4.3 Usability	11
1.5 Constraints	11
1.6 Assumptions and Dependencies	11
1.7 Future Enhancements	12
1.8 Conclusion	12

2 Software Requirements Specification (SRS) for Place Order Management in CUET Food Delivery Service	13
2.1 Introduction	13
2.1.1 Purpose	13
2.1.2 Scope	13
2.2 Overall Description	13
2.2.1 Product Perspective	13
2.2.2 User Classes and Characteristics	13
2.3 Functional Requirements	13
2.3.1 Browse Menus	13
2.3.2 View Food Item Details	13
2.3.3 Add to Cart	14
2.3.4 Place Order	14
2.3.5 Order Tracking	14
2.4 Non-functional Requirements	14
2.4.1 Performance	14
2.4.2 Security	14
2.4.3 Usability	14
2.5 Constraints	14
2.6 Assumptions and Dependencies	15
2.7 Future Enhancements	15
2.8 Conclusion	15
3 Software Requirements Specification (SRS) for Food Management in CUET Food Delivery Service	16
3.1 Introduction	16
3.1.1 Purpose	16
3.1.2 Scope	16

3.2	Overall Description	16
3.2.1	Product Perspective	16
3.2.2	User Classes and Characteristics	16
3.3	Functional Requirements	16
3.3.1	Add Food Item	16
3.3.2	Update Food Item	16
3.3.3	Categorize Food Items	17
3.3.4	Manage Images	17
3.4	Non-functional Requirements	17
3.4.1	Performance	17
3.4.2	Security	17
3.4.3	Usability	17
3.5	Constraints	17
3.6	Assumptions and Dependencies	18
3.7	Future Enhancements	18
3.8	Conclusion	18
4	Use Case Diagram with Graphical and Textual Description	19
5	Activity Diagram	25
6	Swimlane Diagram	27
7	Static Model - Class Diagram	30
8	Dynamic Model - Sequence Diagram	31
9	Safety and Security Requirements	33
9.1	Access Requirements	33
9.1.1	User Authentication and Authorization	33
9.1.2	Delivery Personnel Access	33

9.1.3	Vendor Access	33
9.1.4	Limited Access to Order Information	33
9.2	Integrity Requirements	33
9.2.1	Data Integrity	33
9.2.2	Order Information Integrity	33
9.2.3	System Integrity	34
9.3	Privacy Requirements	34
9.3.1	User Data Protection	34
9.3.2	Compliance with Privacy Laws	34
9.3.3	Customer Consent	34
10	Architecture	35
10.1	Data Centered Architecture	35
10.2	UML communication Diagram	35
10.3	Rationale for choosing architectural model	36
10.3.1	Data-Centered Architecture	36
10.3.2	UML Communication Diagram	37
10.4	Technology, Software & Hardware	37
11	GUI (Graphical User Interface) Design	39
12	Design	44
12.1	Component level design pattern	44
12.1.1	User Management	44
12.1.2	Order Placement	45
12.1.3	Food Management	46
12.2	Dataflow Diagram (DFD)	47
12.2.1	First Level DFD for the System	47
12.2.2	DFD for user management	48
12.2.3	DFD for order management	48

12.2.4 DFD for User Food Management	49
12.3 Entity Relationship Diagram (ERD)	49
13API Documentation Manual	51
13.1 User Management	51
13.1.1 List Users	51
13.1.2 Add User	52
13.1.3 Update	53
14Testing and Sustainability Plan	54
14.1 Requirements/Specifications-based System Level Test Cases . .	54
15Process Model	55
15.1 Process Model used	55
15.2 Rationale	56
16Risk Analysis	58
16.1 Risk identification	58
16.2 Risk projection	58
16.3 Risk Refinement	58
16.4 Risk Mitigation, Monitoring and Management	59
16.4.1 Risk Mitigation	59
16.4.2 Risk Monitoring	59
16.4.3 Risk Management	59
17Constraints to project implementation	61
17.1 Schedule	61
17.2 Software & Hardware Constraints	61
18Hardware and Software Resource (Tools/Lan- guage) Requirements	62
18.1 Software Requirements	62

18.2 Hardware Requirements	62
19Project Timeline and Schedule	63
20Estimated Budget	64
20.1 Type of Software Project:	64
20.2 Size of Software Project:	64
20.3 Development Team Size:	64
20.4 Other Considerations:	64
21Social, Cultural Environmental impact	66
21.1 Social impact	66
21.2 Cultural impact	66
21.3 Environmental impact	66
References	67

List of Figures

4.1 User Module Use Case Diagram.	20
4.2 Place Food Order Module Use Case Diagram.	22
4.3 Add Food Item Module Use Case Diagram.	24
5.1 Activity Diagram: User Module	25
5.2 Activity Diagram: Place Order	26
5.3 Activity Diagram: Food Management	26
6.1 Swim-lane Diagrams: User Module	27
6.2 Swim-lane Diagrams: Place Order	28
6.3 Swim-lane Diagrams: Food Management	29
7.1 Class diagram.	30

8.1	Sequence diagram : User Module	31
8.2	Sequence diagram: Place Order	32
8.3	Sequence diagram: Food Management	32
10.1	Data Centered Architecture	35
10.2	UML communication Diagram	36
11.1	Splash	39
11.2	Login or Creat Account	39
11.3	Login Page	40
11.4	Sign Up	40
11.5	Confirm Mobile Number using OTP	40
11.6	User Profile	40
11.7	Home Page	41
11.8	Food Details	41
11.9	Order Details	42
11.10	Checkout Information	42
11.11	Order History and Other Informations	42
11.12	About Us	42
11.13	Order Confirmation Message	43
12.1	Structure Chart for CUET food delivery app	44
12.2	Component level design for user management.	45
12.3	Component level design for Order Management.	46
12.4	Component level design for user food management.	47
12.5	First Level Dataflow diagram for CUET Food delivery app . . .	47
12.6	Dataflow diagram for user management	48
12.7	Dataflow diagram for order Management.	49
12.8	Dataflow diagram for Food management	49
12.9	ER Diagram for food delivery system	50
13.1	Header and Request URL	51
13.2	Response Body and response header	52

13.3	Add User	52
13.4	Response of Add USER	52
13.5	INPUT Parameters for the request JSON.	53
13.6	Header and Request URL	53
14.1	Requirements/Specifications-based System Level Test Cases (CUET Food Delivery app)	54
15.1	Prototyping Process Model	56
19.1	Project Gantt Chart	63

1. Software Requirements Specification (SRS) for User Management in CUET Food Delivery Service

1.1 Introduction

1.1.1 Purpose

The purpose of this document is to provide a detailed description of the requirements for the "User Module" [1] of the CUET Food Delivery Service. This module is responsible for managing user accounts, preferences, and authentication.

1.1.2 Scope

The "User Module" will be a fundamental component of the CUET Food Delivery Service, enabling users to create accounts, manage profiles, and track order history. It interacts with other modules such as the Order Food Module and Food Management Module.

1.2 Overall Description

1.2.1 Product Perspective

The "User Module" closely interacts with the Order Food Module, Food Management Module, and handles user data securely in the database.

1.2.2 User Classes and Characteristics

- Customers: Users who browse and place orders.
- Vendors: Restaurant owners managing their food listings.
- Delivery Personnel: Users responsible for delivering orders.

1.3 Functional Requirements

There are some functional requirements that need to be mentioned.

1.3.1 User Registration

- The system shall provide a registration form for users to create accounts.
- Users must provide necessary details, including name, contact information, and address.
- The system shall validate and ensure the completeness of user registration details.
- Upon successful registration, the system shall send a confirmation email or SMS to the user.

1.3.2 User Authentication

- The system shall authenticate users during login using secure methods.
- Users should be able to reset their passwords through a secure process.

1.3.3 Profile Management

- Users shall be able to update their profiles, including contact details and delivery addresses.
- The system shall allow users to link their accounts to social media for easy login.

1.3.4 Order History

- The system shall maintain a record of users' order history.
- Users should be able to view and reorder from their order history.

1.4 Non-functional Requirements

1.4.1 Performance

- The system should support a large number of user registrations and logins simultaneously.
- User registration and login processes should be completed within 5 seconds.

1.4.2 Security

- User information must be securely stored.
- All user interactions must be encrypted and transmitted over HTTPS.

1.4.3 Usability

- The user interface for registration and login should be intuitive.
- Error messages should be clear and user-friendly.

1.5 Constraints

- The system must be developed using Flutter, Restful Api, Laravel.
- The system must run on mobile and desktop.

1.6 Assumptions and Dependencies

- The system assumes that users have access to the internet and a compatible device.

- The system depends on the accuracy of user information provided during registration.

1.7 Future Enhancements

- The system could be enhanced to provide personalized recommendations based on order history.
- Integration with third-party authentication services for user convenience.

1.8 Conclusion

This Software Requirements Specification outlines the detailed requirements for the "User Module" of the CUET Food Delivery Service. Successful implementation will result in a seamless and secure user experience.

2. Software Requirements Specification (SRS) for Place Order Management in CUET Food Delivery Service

2.1 Introduction

2.1.1 Purpose

The purpose of this document is to provide a detailed description of the requirements for the "Place Order Module" of the CUET Food Delivery Service. This module is responsible for handling user interactions related to browsing, selecting, and placing orders for food items from various vendors.

2.1.2 Scope

The "Place Order Module" is a vital component of the CUET Food Delivery Service, facilitating users in exploring available food options, customizing orders, and completing transactions. It interacts with the User Module and Food Management Module.

2.2 Overall Description

2.2.1 Product Perspective

The "Place Order Module" interacts closely with the User Module, enabling users to browse menus, add items to their cart, and place orders. It relies on the Food Management Module to retrieve and display food item details.

2.2.2 User Classes and Characteristics

- Customers: Users who browse menus, customize orders, and place food orders.

2.3 Functional Requirements

2.3.1 Browse Menus

- The system shall display a list of available restaurants and their menus.
- Users should be able to filter and search for specific cuisines or dishes.

2.3.2 View Food Item Details

- Users shall be able to view detailed information about each food item, including name, description, price, and images.
- The system shall display nutritional information, allergens, and preparation time when available.

2.3.3 Add to Cart

- Users shall be able to add food items to their shopping cart.
- The system shall calculate the total order amount dynamically as items are added.

Cart Management

- Users should be able to view and edit the contents of their shopping cart.
- The system shall update the total order amount when items are added or removed from the cart.

2.3.4 Place Order

- The system shall prompt users to confirm and place their orders.
- Users should receive an order confirmation notification via email or SMS.

2.3.5 Order Tracking

- Users shall be able to track the status of their orders in real-time.
- The system shall update users on order preparation, dispatch, and estimated delivery time.

2.4 Non-functional Requirements

2.4.1 Performance

- The system should support a large number of simultaneous users browsing menus and placing orders.
- Menu loading and order placement processes should be completed within 5 seconds.

2.4.2 Security

- User payment information must be securely handled and processed.
- All order-related interactions must be transmitted over HTTPS.

2.4.3 Usability

- The user interface for browsing menus and placing orders should be intuitive.
- Error messages should be clear and guide users in case of issues.

2.5 Constraints

- The system must integrate with secure payment gateways for transaction processing.
- The system must comply with relevant food safety and hygiene regulations.

2.6 Assumptions and Dependencies

- The system assumes that users have registered accounts through the User Module
- Order information depends on the accuracy of the menu and item details provided by vendors.

2.7 Future Enhancements

- Integration with loyalty programs for customer rewards.
- Advanced customization options for food items.

2.8 Conclusion

This Software Requirements Specification outlines the detailed requirements for the "Place Order Module" of the CUET Food Delivery Service. Successful implementation will result in a seamless and efficient food ordering experience for users.

3. Software Requirements Specification (SRS) for Food Management in CUET Food Delivery Service

3.1 Introduction

3.1.1 Purpose

The purpose of this document is to provide a detailed description of the requirements for the "Food Management Module" of the CUET Food Delivery Service. This module is responsible for managing food-related information, including creating, updating, and categorizing food items.

3.1.2 Scope

The "Food Management Module" is a pivotal part of the CUET Food Delivery Service, allowing vendors to add, edit, and organize their food listings. It interacts with the Order Food Module to provide real-time menu information.

3.2 Overall Description

3.2.1 Product Perspective

The "Food Management Module" closely interacts with the Order Food Module, supplying information on available food items, prices, and descriptions. It relies on the User Module for authentication and authorization.

3.2.2 User Classes and Characteristics

- Vendors: Users who manage their restaurant's menu and food listings.

3.3 Functional Requirements

3.3.1 Add Food Item

- Vendors shall be able to add new food items to their menu.
- The system shall require vendors to provide details such as item name, description, price, and images.
- The system shall validate and ensure the completeness of food item details provided by vendors.
- Upon successful addition, the system shall notify the vendor of the new item.

3.3.2 Update Food Item

- Vendors shall be able to edit existing food items.

- The system shall allow vendors to modify details such as name, description, price, and images.
- The system shall validate and ensure the completeness of updated food item details.
- Upon successful update, the system shall notify the vendor of the changes.

3.3.3 Categorize Food Items

- Vendors shall be able to categorize food items into different menus or sections.
- The system shall allow vendors to create, edit, and delete menu categories.

Availability Status

- Vendors shall be able to set the availability status of each food item (e.g., available, sold out).
- The system shall update the availability status in real-time for users browsing the menu

3.3.4 Manage Images

- Vendors shall be able to upload and manage images for each food item.
- The system shall display these images along with the food item details in the menu.

3.4 Non-functional Requirements

3.4.1 Performance

- The system should support a large number of simultaneous updates to food item details.
- Food item addition and update processes should be completed within 5 seconds.

3.4.2 Security

- Food item and vendor information must be securely stored.
- All interactions with the module must be authenticated and authorized.

3.4.3 Usability

- The user interface for adding, updating, and categorizing food items should be intuitive.
- Error messages should be clear and assist vendors in case of issues.

3.5 Constraints

- The system must be developed using Flutter, Laravel, Rest Api.

- The system must comply with relevant food safety and hygiene regulations.

3.6 Assumptions and Dependencies

- The system assumes that vendors have registered accounts through the User Module.
- Food item details depend on the accuracy provided by vendors.

3.7 Future Enhancements

- Integration with inventory management for real-time tracking.
- Analytics for vendors to track the popularity of food items.

3.8 Conclusion

This Software Requirements Specification outlines the detailed requirements for the "Food Management Module" of the CUET Food Delivery Service. Successful implementation will result in an organized and efficient management system for vendors to handle their food listings.

4. Use Case Diagram with Graphical and Textual Description

Use Case:

User Module [2] for CUET Food Delivery Service

Iteration:

1

Primary Actor:

Student

Goal in Context:

To create a new account for accessing the CUET Food Delivery Service.

Preconditions:

- The CUET Food Delivery Service must be installed on the user's device.
- The user has not registered an account before.

Trigger:

The user decides to create a new account.

Scenario:

1. The user opens the CUET Food Delivery Service.
2. The user selects the option to register a new account.
3. The system prompts the user to provide necessary details, including name, contact information, and a preferred password.
4. The system validates the entered information and registers the new account.
5. The system sends a confirmation email or SMS to the user.

Exceptions:

- If the entered information is incomplete or invalid, the system prompts the user to correct it.
- If there is a technical failure during the registration process, the system informs the user and suggests trying again later.

Priority:

High priority, as it is a fundamental step for accessing the app.

When Available:

First increment.

Frequency of Use:

Frequent

Channel to Actor:

Via a mobile app on a smartphone using an internet connection.

Secondary Actors:

System Administrator

Channels to Secondary Actors:

- System Administrator: Mobile based system.

Open Issues:

- Ensuring secure storage of user registration information.
- Implementing measures to prevent unauthorized access.

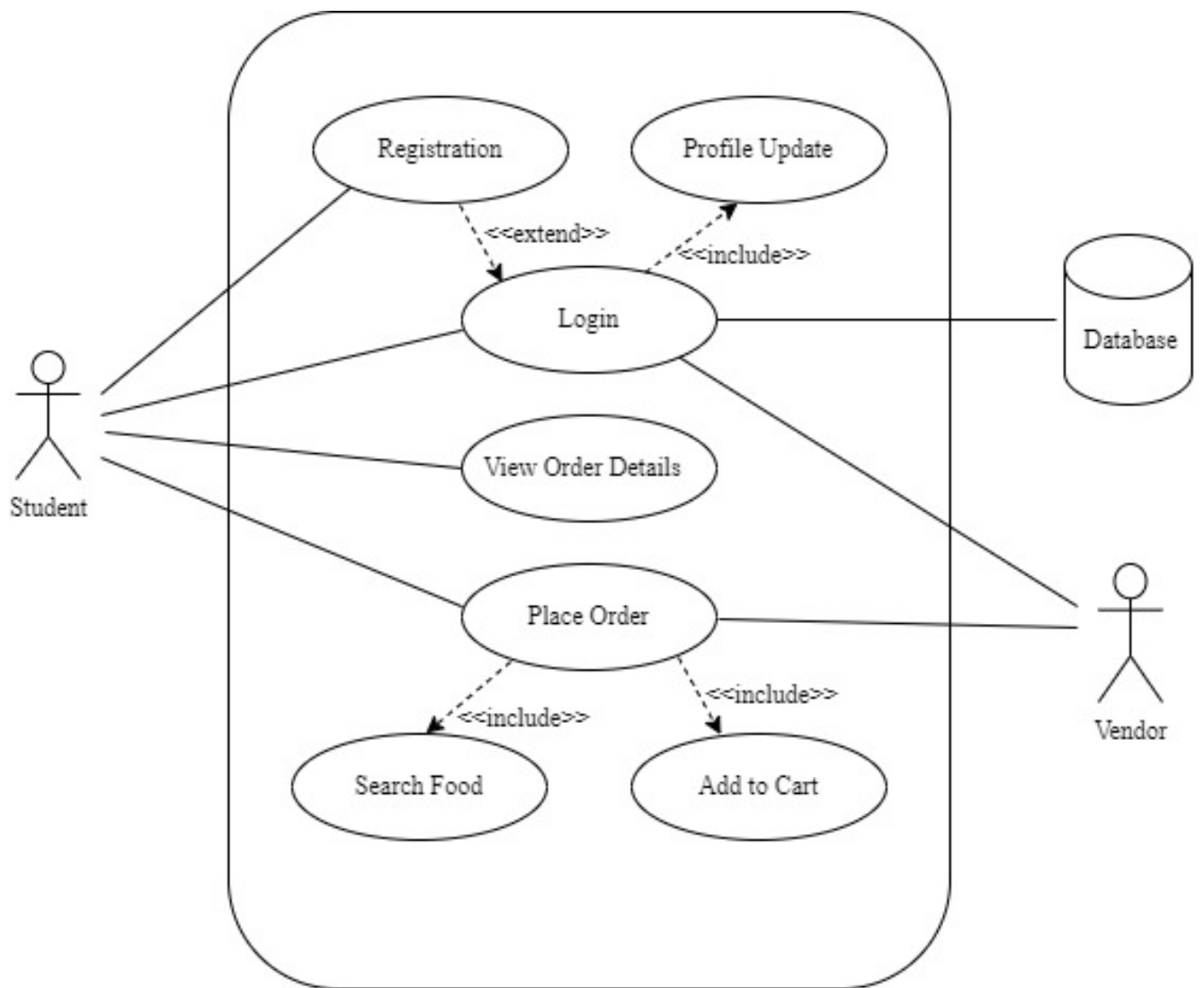


Figure 4.1: User Module Use Case Diagram.

Use Case:

Place Food Order for CUET Food Delivery Service

Iteration:

1

Primary Actor:

Student

Goal in Context:

To place an order for food from campus vendors using the CUET Food Delivery Service.

Preconditions:

- The CUET Food Delivery Service must be installed on the user's device.
- The user must have a registered account with appropriate user ID and password.
- The campus vendors must be registered and active on the app.

Trigger:

The user decides to order food from a campus vendor.

Scenario:

1. The user opens the CUET Food Delivery Service and logs in using their user ID and password.
2. The system validates the credentials and displays the home screen.
3. The user browses the list of available vendors and selects a specific vendor to view their menu.
4. The user adds food items to their cart.
5. The user confirms the order and selects a delivery address.
6. The system confirms the order and provides an estimated delivery time.
7. The vendor prepares the order for delivery.
8. The delivery person delivers the food to the specified address.
9. The user makes the payment using a cash on delivery method.

10. The system updates the order status to "Delivered."

Exceptions:

- If the user's credentials are incorrect, the system displays an error message and prompts for valid credentials.
- If there is a technical failure during the order process, the system informs the user and suggests trying again later.
- If a selected food item is unavailable, the system notifies the user and allows them to make a different selection

Priority:

High priority, as it constitutes the core functionality of the Food Delivery App.

When Available:

First increment.

Frequency of Use:

Frequent

Channel to Actor:

Via a mobile app on a smartphone using an internet connection.

Secondary Actors:

Campus Vendor, Delivery Person

Channels to Secondary Actors:

- Campus Vendor: Receives order details and confirmation through the app.
- Delivery Person: Receives order details and real-time tracking information through the app.

Open Issues:

- Security measures to protect user payment information.
- Ensuring accurate and up-to-date vendor menus.
- Implementing efficient real-time order tracking for the user.

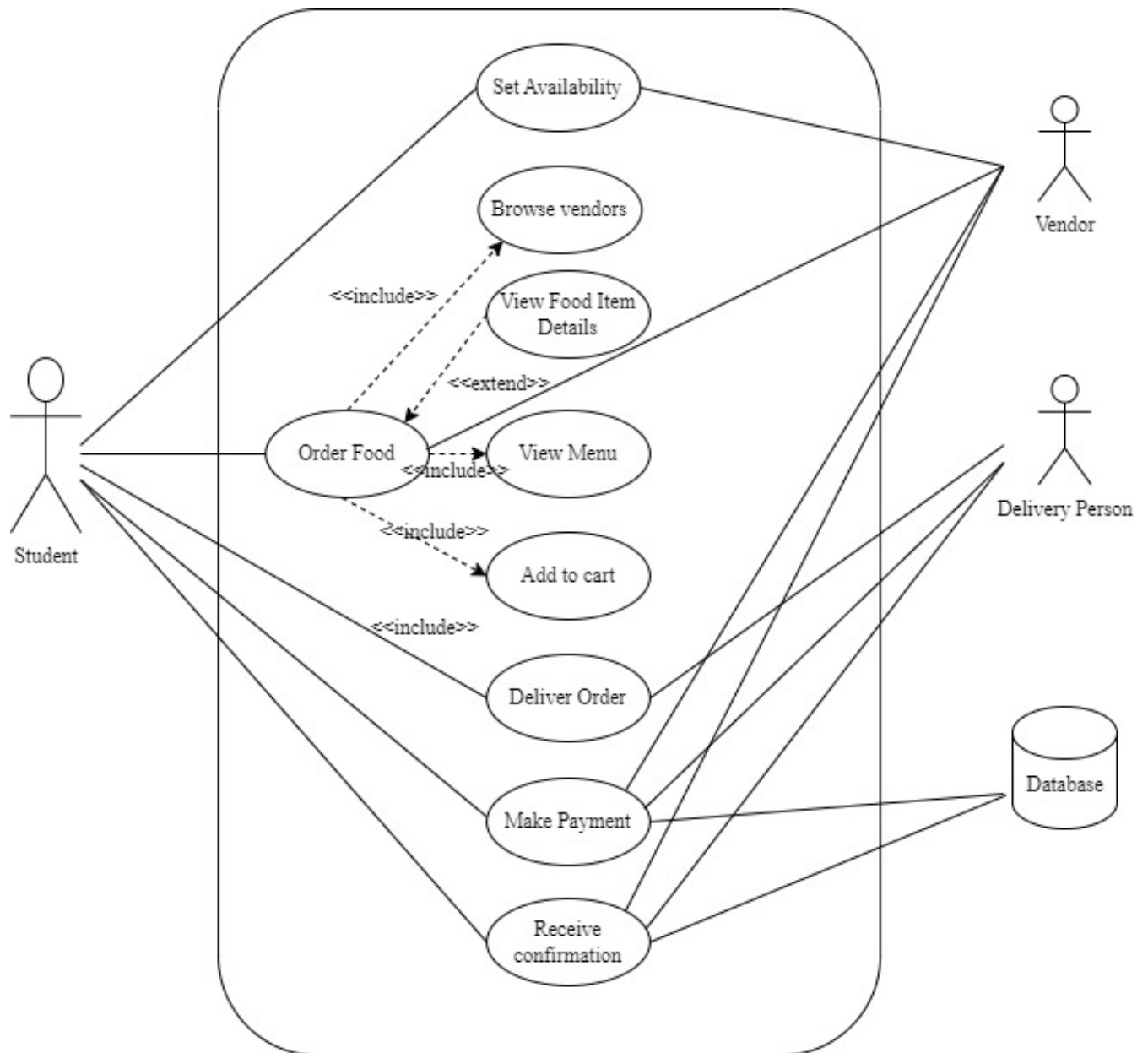


Figure 4.2: Place Food Order Module Use Case Diagram.

Use Case:

Add Food Item for CUET Food Delivery Service

Iteration:

1

Primary Actor:

Vendor

Goal in Context:

To add a new food item to the menu using the CUET Food Delivery Service.

Preconditions:

- The CUET Food Delivery Service must be installed on the vendor's device.
- The vendor must have a registered account with appropriate user ID and password.

Trigger:

The vendor decides to add a new food item to their menu.

Scenario:

1. The vendor opens the CUET Food Delivery Service.
2. The vendor logs in using their user ID and password.
3. The system validates the credentials and displays the vendor dashboard.
4. The vendor selects the option to add a new food item.
5. The system prompts the vendor to provide details, including item name, description, price, and images.
6. The system validates the entered information and adds the new food item to the menu.
7. The system notifies the vendor of the successful addition.
8. The vendor updates the situation of order.

9. The vendor provides the food to the delivery person.

10. The vendor confirms the delivery after receiving information from the delivery person.

Exceptions:

- If the entered information is incomplete or invalid, the system prompts the vendor to correct it.
- If there is a technical failure during the addition process, the system informs the vendor and suggests trying again later.

Priority:

High priority, as it is a fundamental step for managing the menu.

When Available:

First increment.

Frequency of Use:

Frequent

Channel to Actor:

Via a mobile app on a smartphone using an internet connection.

Secondary Actors:

Student, Delivery Person

Channels to Secondary Actors:

- Student: Receives real-time tracking of orders through the app.
- Delivery Person: Receives order details and real-time tracking information through the app.

Open Issues:

- Ensuring secure storage of menu information.
- Implementing measures to prevent unauthorized access to menu management functionalities.

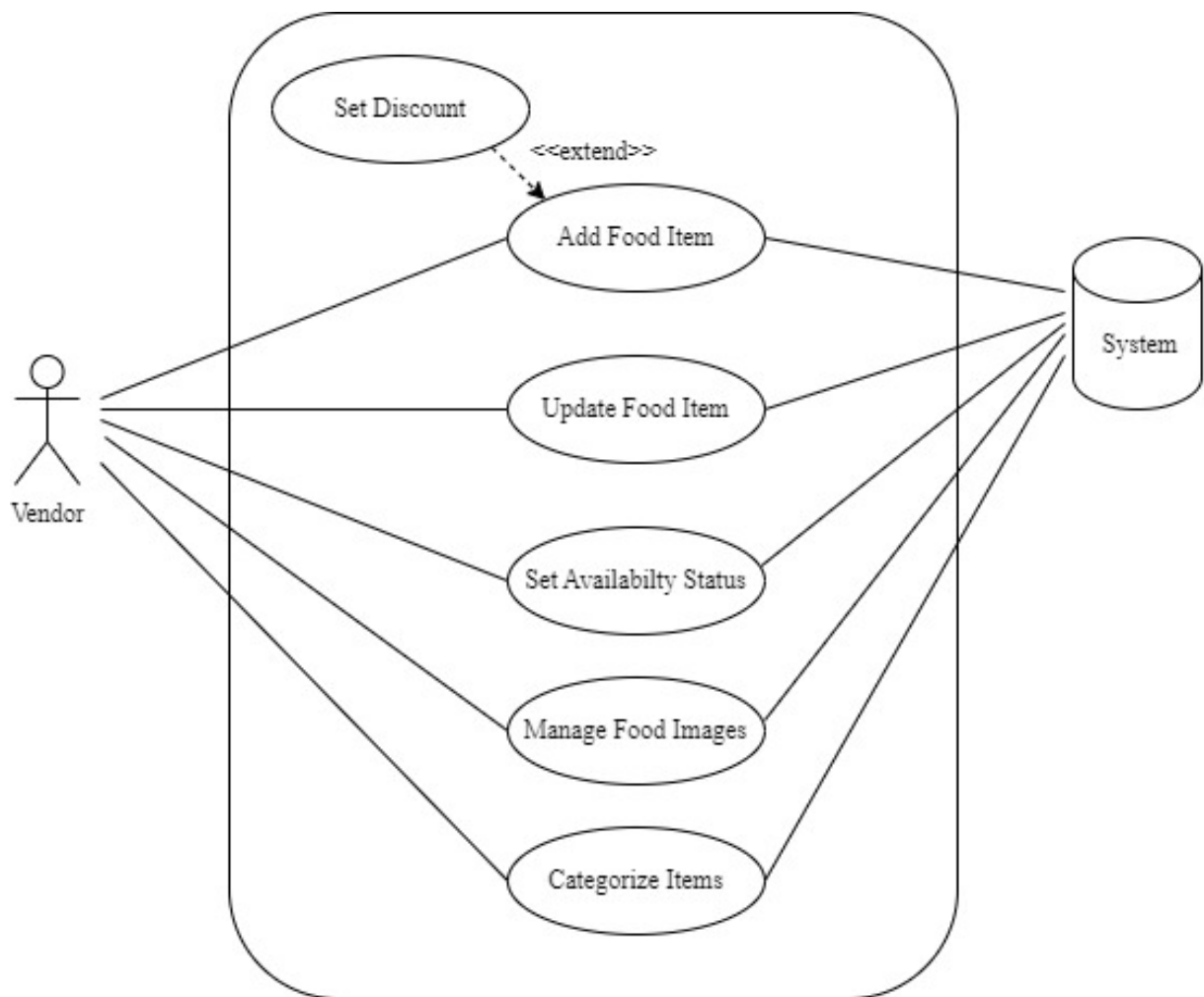


Figure 4.3: Add Food Item Module Use Case Diagram.

5. Activity Diagram

An activity diagram [3] is a type of UML (Unified Modeling Language) diagram that illustrates the dynamic aspects of a system by modeling the flow of activities. It is particularly useful for visualizing business processes and workflows, showing the sequence of activities or actions within a system or between multiple entities.

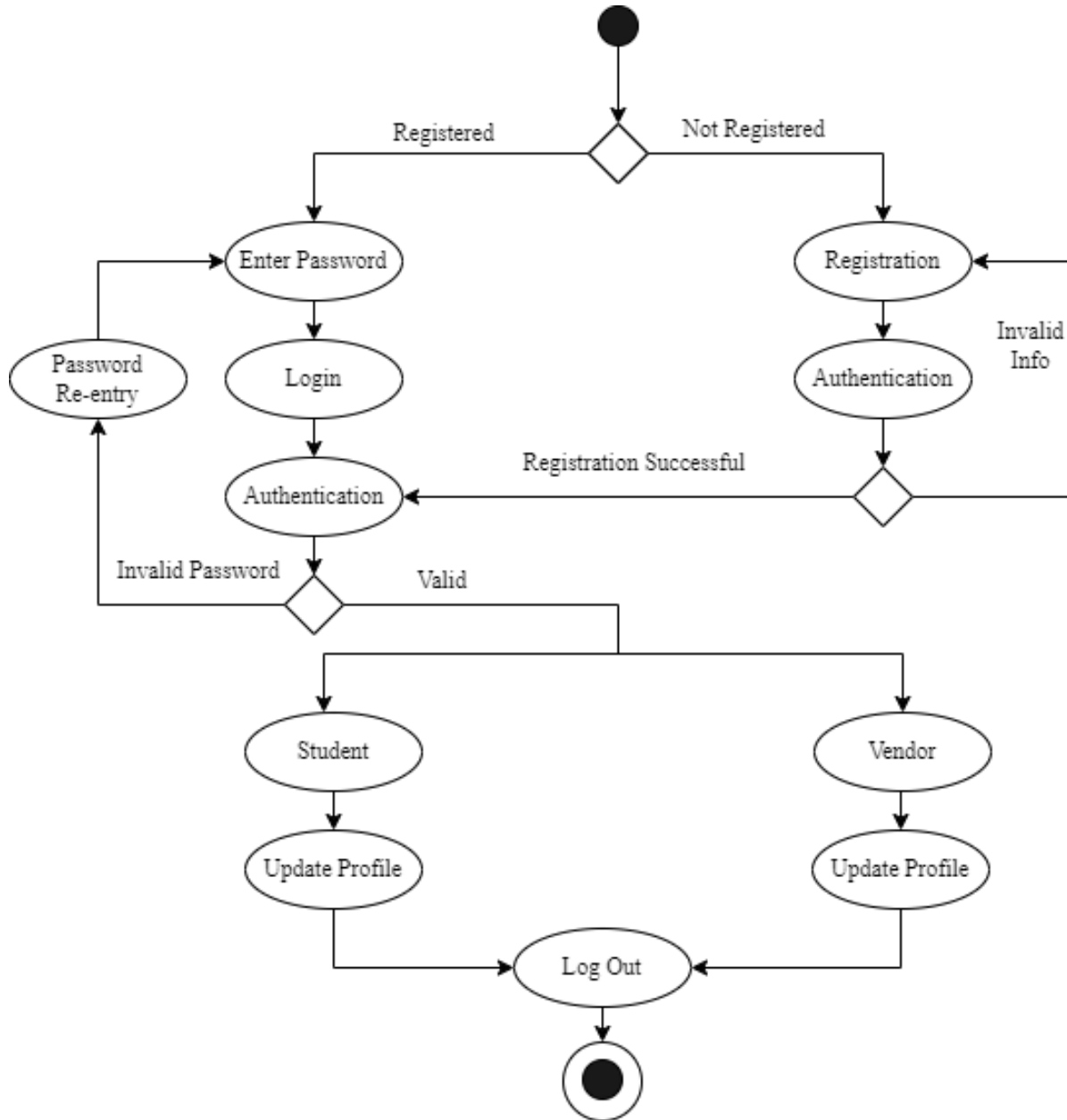


Figure 5.1: Activity Diagram: User Module

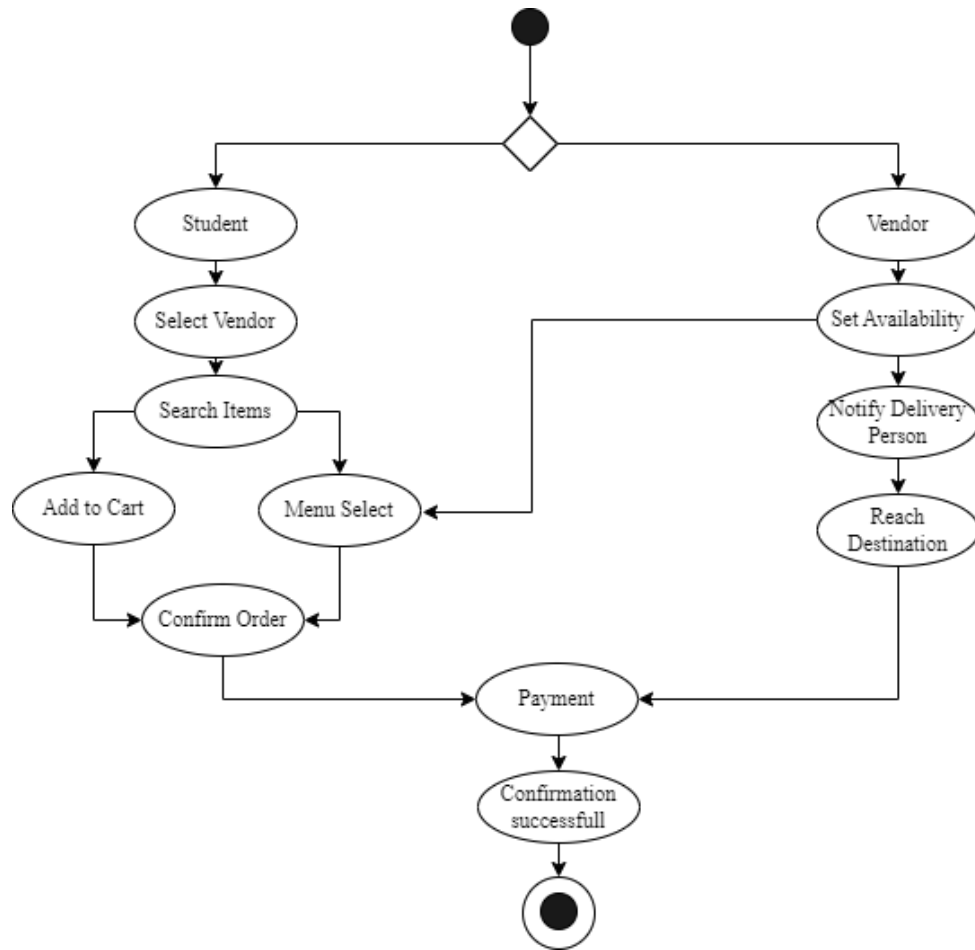


Figure 5.2: Activity Diagram: Place Order

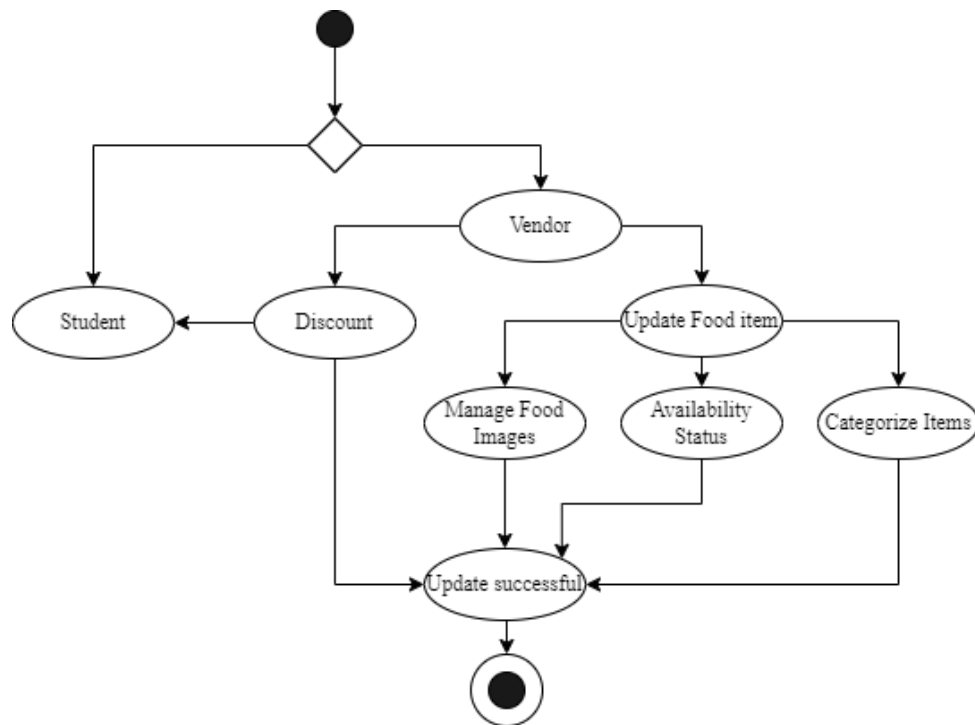


Figure 5.3: Activity Diagram: Food Management

6. Swimlane Diagram

Swimlane diagrams [4] are flowcharts that show a process from start to finish. These diagrams also show who is responsible for each step in the process. Much like a swimming pool with established lanes for each swimmer, a swim lane diagram has horizontal or vertical lanes belonging to each person involved in the process.

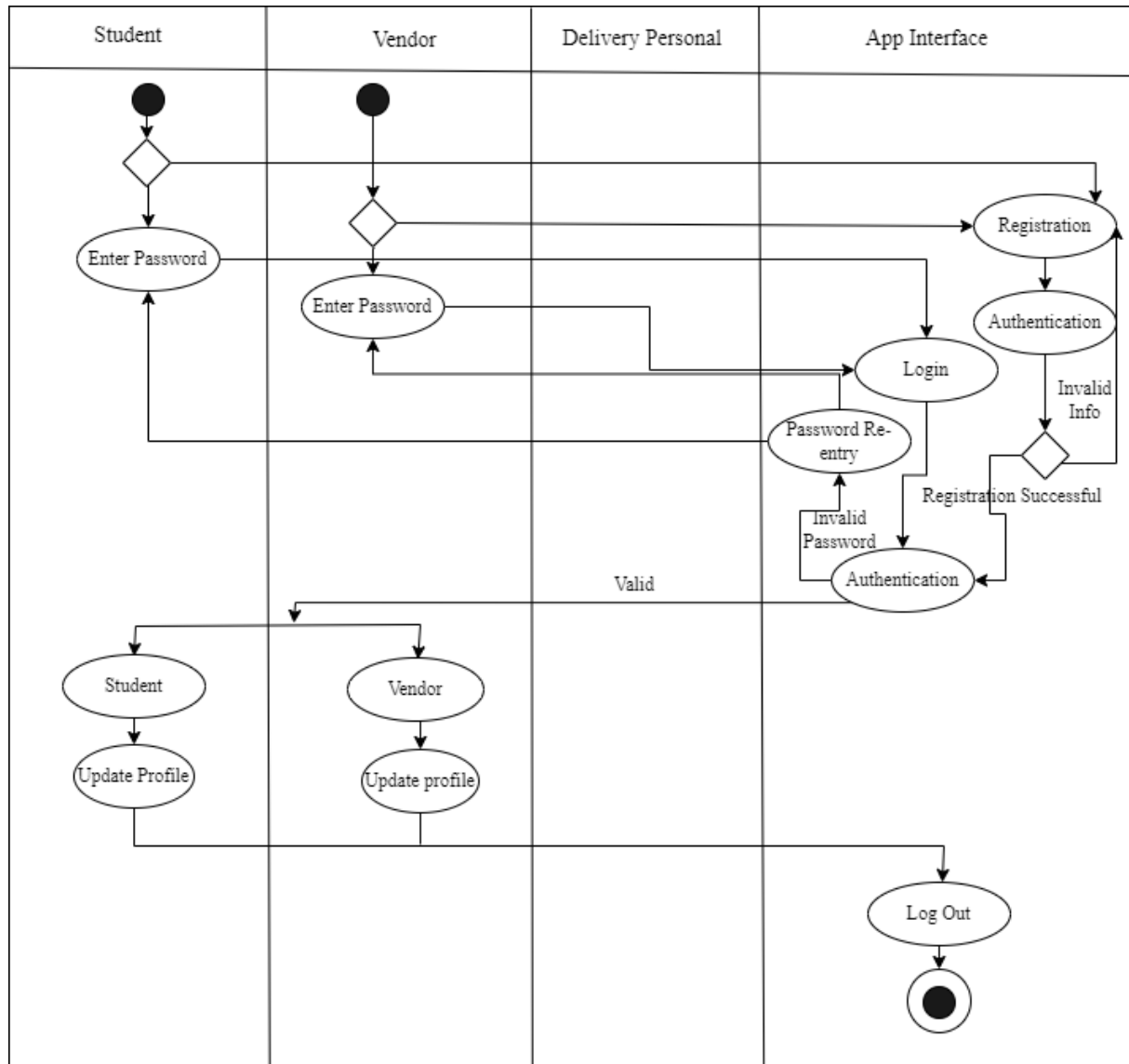


Figure 6.1: Swim-lane Diagrams: User Module

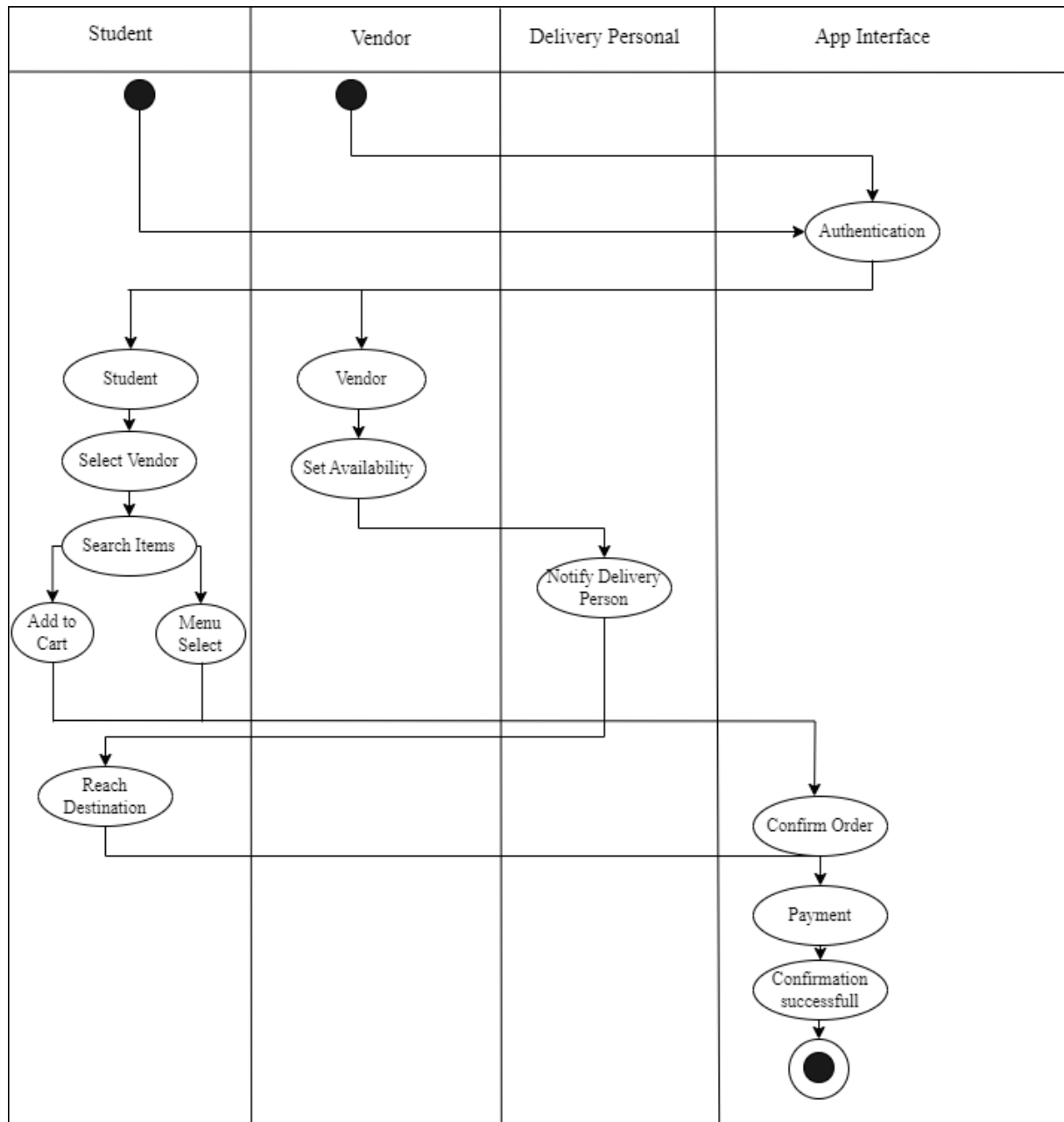


Figure 6.2: Swim-lane Diagrams: Place Order

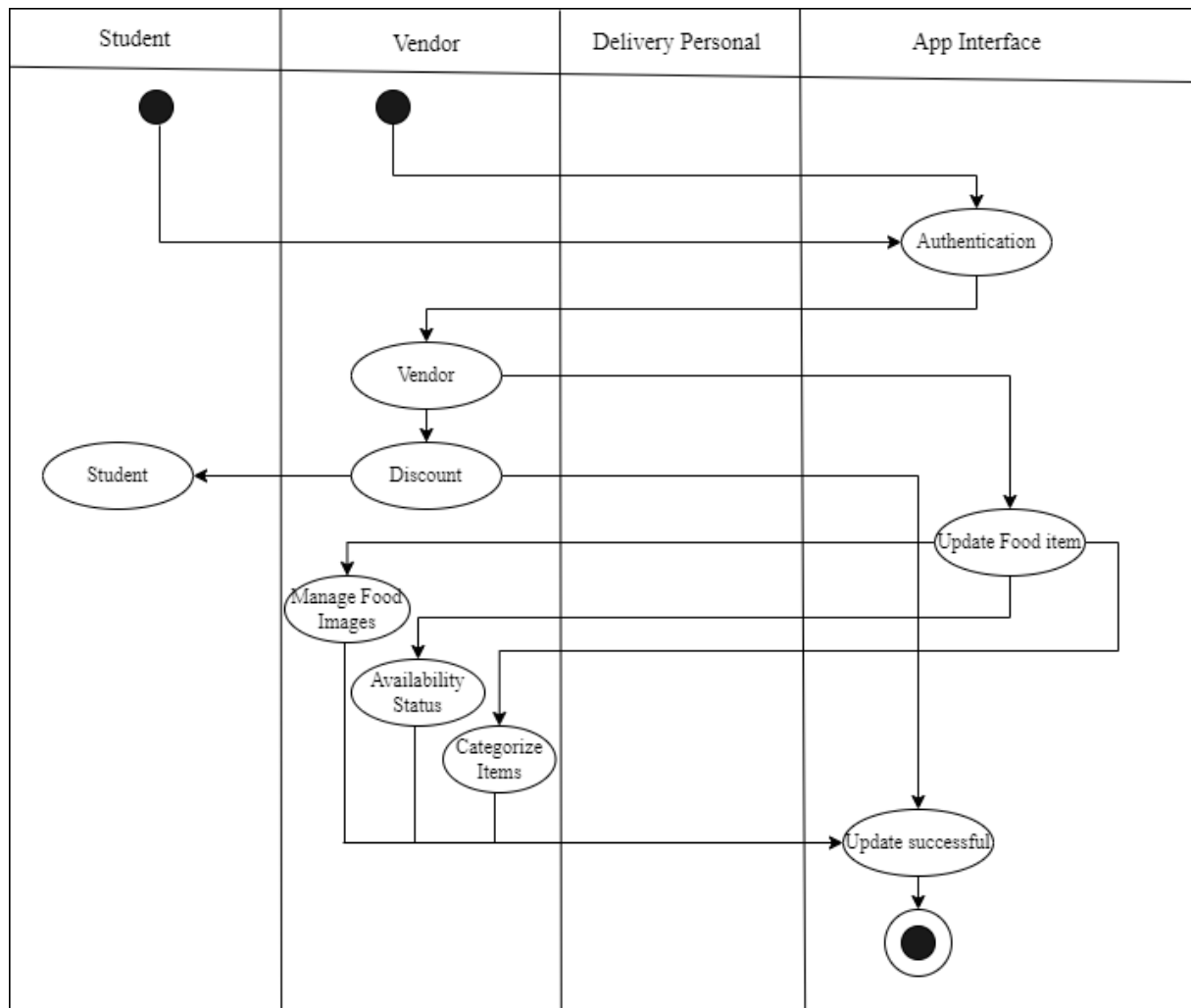


Figure 6.3: Swim-lane Diagrams: Food Management

7. Static Model - Class Diagram

A class diagram [5] for a food delivery app involves identifying the main classes, their attributes, and their relationships. Below is a simplified example of a class diagram for a food delivery app:

The arrows indicate relationships between classes. For example:

- A student can place an order, so there is an association between the Student class and the Order class.
- An order can contain multiple items, so there is an association between the Order class and the OrderItem class.
- Each order is associated with a specific campus vendor, indicating the relationship between Order and CampusVendor.

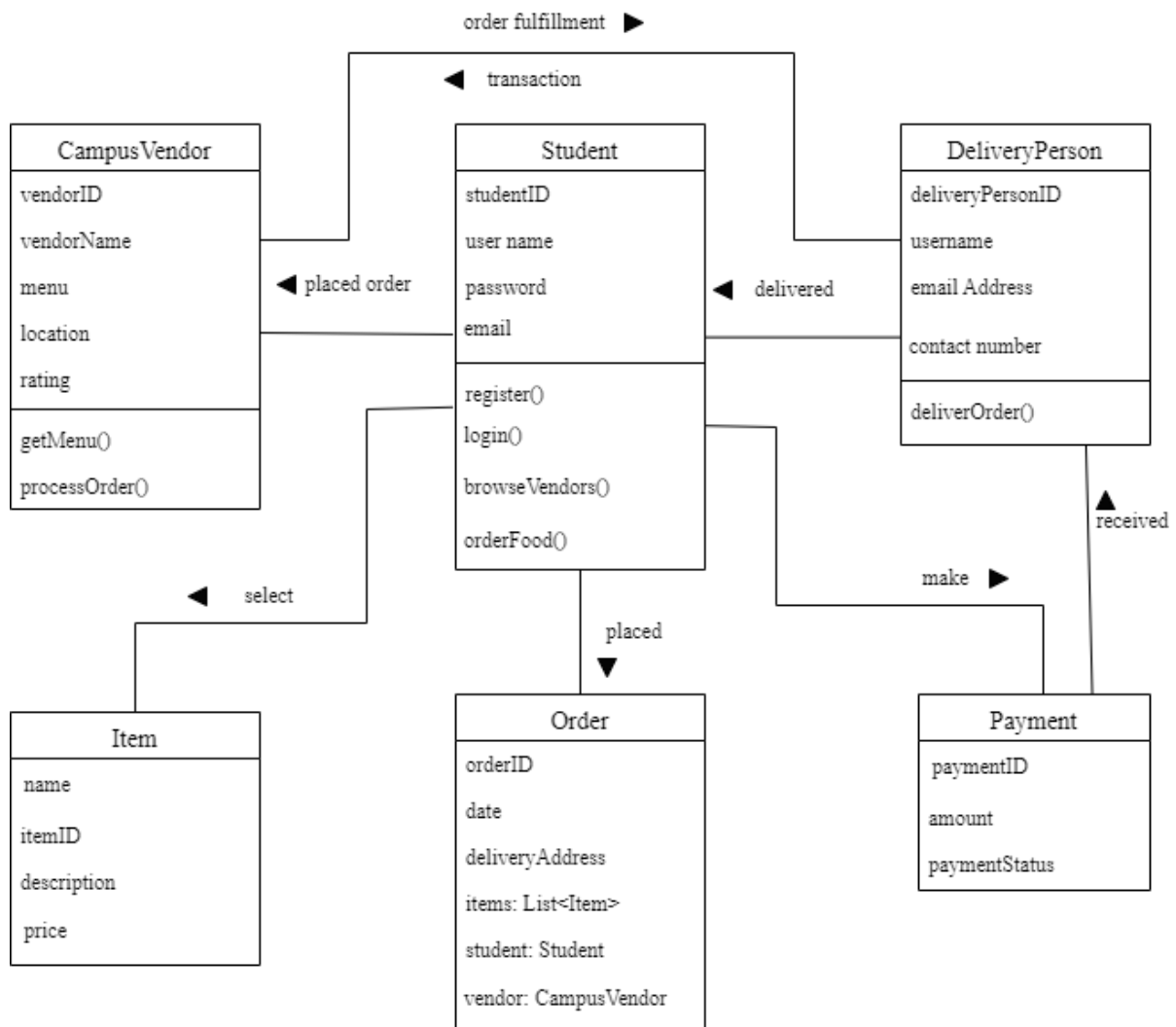


Figure 7.1: Class diagram.

8. Dynamic Model - Sequence Diagram

Utilining the interactions between the main actors (Student, Campus Vendor, Delivery Person) and the system components [6]. Please note that this is a simplified example, and in a real-world scenario, there may be additional details and interactions.

The CUET Food Delivery mobile app encompasses various crucial functionalities catering to user interaction, order processing, and efficient food management. Here's a brief overview of each:

User Module: This component serves as the gateway for users to access the CUET Food Delivery Service. It includes features like account creation, login, profile management, and settings. Users can register, log in securely, manage their personal details, and customize their preferences within this module.

Place Order: This functionality allows users to browse through a variety of restaurants, cuisines, and menu items available within the app. Users can explore, select, and place orders for their desired food items from different eateries listed on the platform. The process involves choosing items, specifying delivery details, and completing the order placement seamlessly.

Food Management & Sequencing: This segment is responsible for orchestrating the operational flow within the app. It manages order processing, dispatch, and delivery sequencing. It involves coordination between restaurants, delivery partners, and users to ensure timely and accurate delivery of orders. This segment optimizes the handling of orders, manages kitchen workflows, and schedules deliveries efficiently.

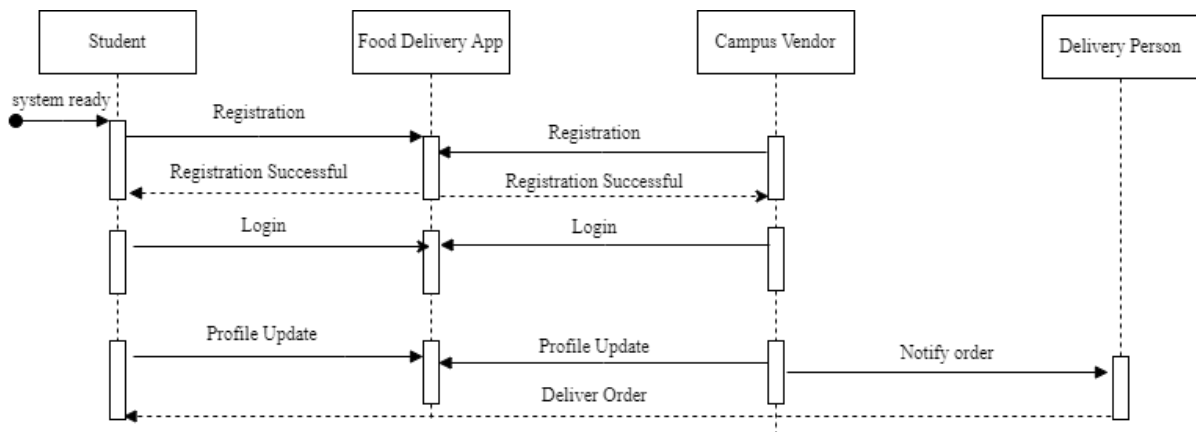


Figure 8.1: Sequence diagram : User Module

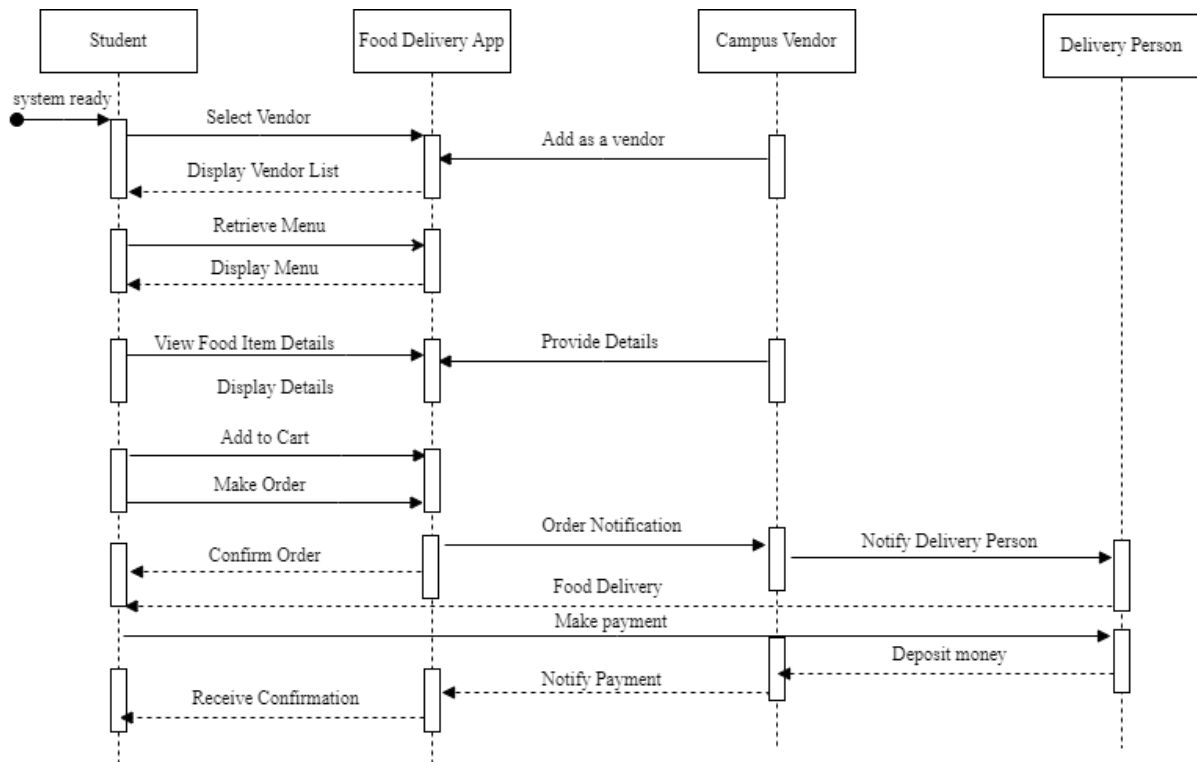


Figure 8.2: Sequence diagram: Place Order

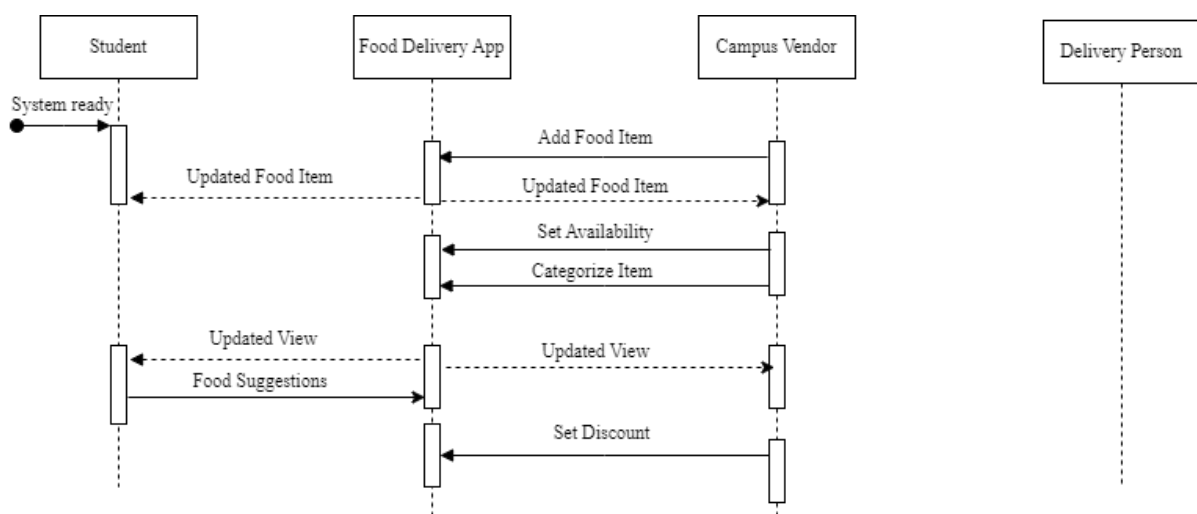


Figure 8.3: Sequence diagram: Food Management

9. Safety and Security Requirements

Ensuring the safety and security of users, their data, and the overall system is paramount in the development and operation of the Campus Food Delivery app. The following requirements are established to address safety and security concerns comprehensively:

9.1 Access Requirements

9.1.1 User Authentication and Authorization

- Only registered users with valid credentials can access the app.
- Access to sensitive functionalities, such as order processing and user profiles, is restricted to authenticated users.

9.1.2 Delivery Personnel Access

- Access to delivery-related functionalities is granted to authorized delivery personnel only.
- Delivery personnel must undergo a verification process before being granted access to order details and customer addresses.

9.1.3 Vendor Access

- Vendors have access to the app's order management system, limited to their respective orders and menu configurations.
- Vendor authentication is required for accessing and updating their information.

9.1.4 Limited Access to Order Information

- Access to detailed order information is restricted to authorized personnel involved in the order fulfillment process.
- Users have access only to their own order history and account details.

9.2 Integrity Requirements

9.2.1 Data Integrity

- Ensure the integrity of user data, including order details and user profiles, during transmission and storage.
- Implement mechanisms to detect and rectify any data corruption or tampering.

9.2.2 Order Information Integrity

- Maintain the integrity of order information from the point of order placement to delivery confirmation.

- Prevent unauthorized modifications to order status and details.

9.2.3 System Integrity

- Regularly perform system checks to identify and address any vulnerabilities or integrity issues within the app.
- Implement version control and change management to maintain the integrity of the overall system.

9.3 Privacy Requirements

9.3.1 User Data Protection

- Encrypt all user data, including personal information and order history, to protect it from unauthorized access.
- Limit access to user data to only essential personnel involved in order processing.

9.3.2 Compliance with Privacy Laws

- Ensure strict adherence to data privacy laws and regulations applicable to the regions where the app operates.
- Regularly update privacy policies and practices to align with evolving legal requirements.

9.3.3 Customer Consent

- Obtain explicit consent from users regarding the storage and processing of their personal information.
- Provide users with options to manage their privacy settings within the app.

By incorporating these safety and security requirements into the development and operational processes, the Campus Food Delivery app aims to provide a secure and trustworthy platform for all stakeholders involved. Regular security audits and updates will be conducted to adapt to evolving threats and maintain a high level of protection.

10. Architecture

10.1 Data Centered Architecture

A data store will reside at the center of this architecture [7][8] and is accessed frequently by the other components that update, add, delete or modify the data present within the store. This figure illustrates a typical data centered style. The client software access a central repository. Variation of this approach are used to transform the repository into a blackboard when data related to client or data of interest for the client change the notifications to client software.

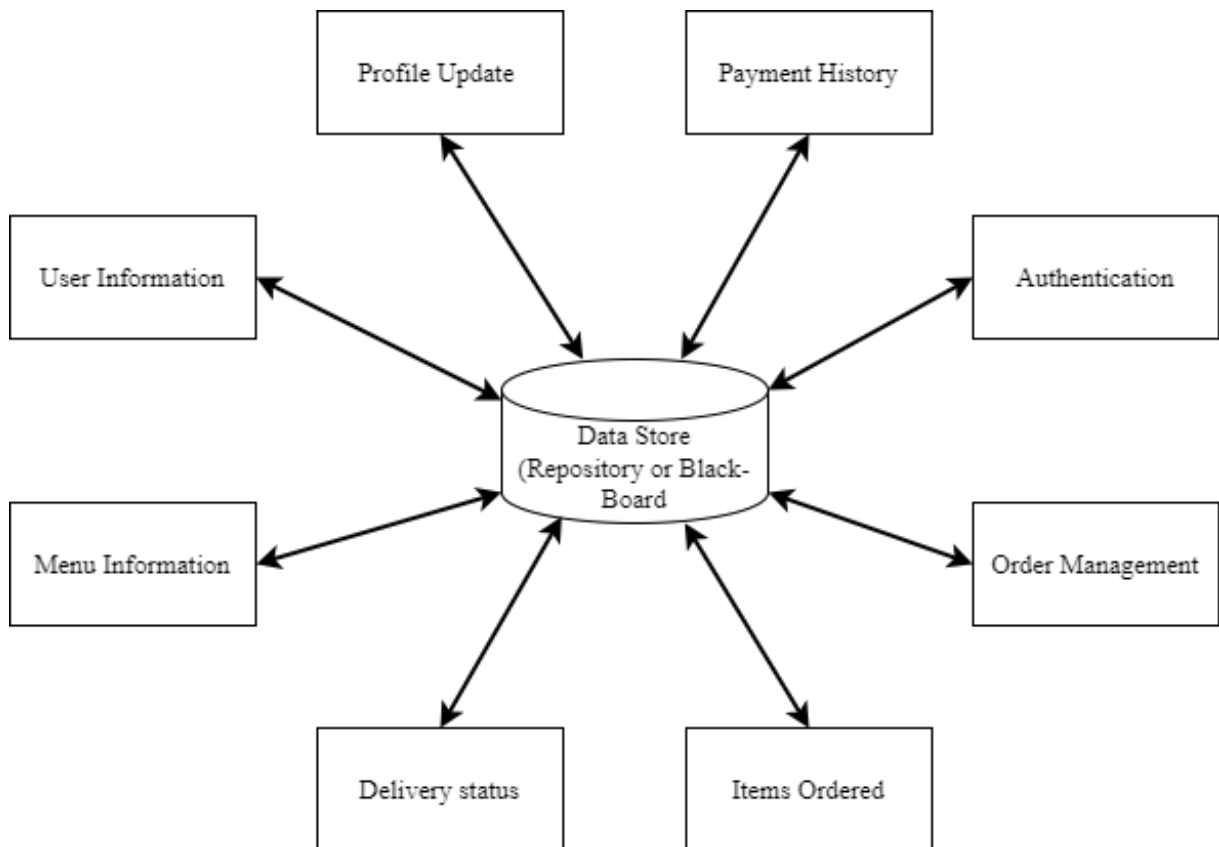


Figure 10.1: Data Centered Architecture

10.2 UML communication Diagram

A communication diagram in the Unified Modeling Language [9] refers to a chart that represents the flow of messages in a system. In a nutshell, it shows how parts of a system interact or in this case, communicate with each other.

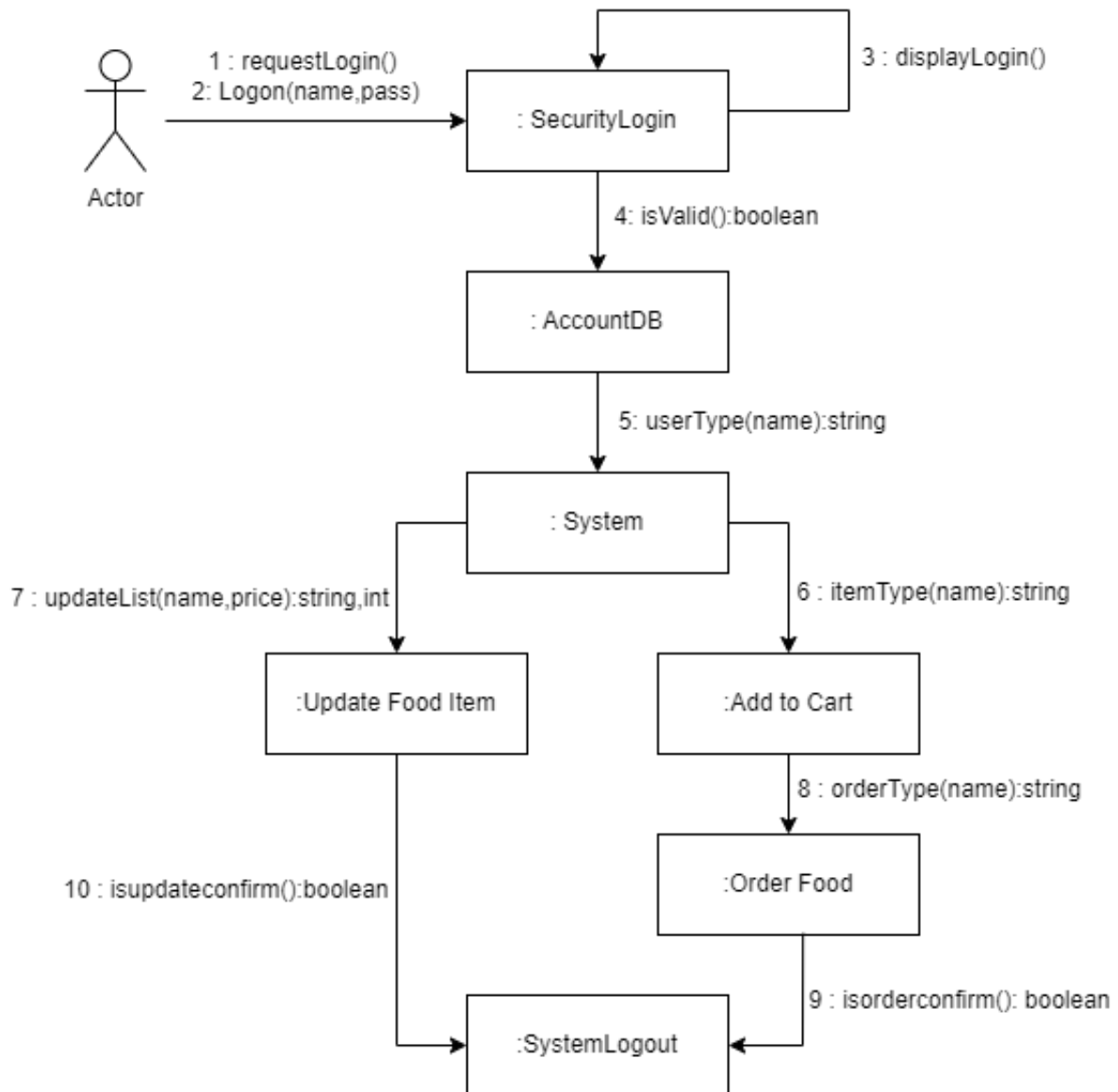


Figure 10.2: UML communication Diagram

10.3 Rationale for choosing architectural model

10.3.1 Data-Centered Architecture

Advantages

- **Scalability:** Data-centered architecture scales effortlessly to accommodate growing user base and data volume.
- **Flexibility:** Easily adapts to new features and functionalities by manipulating data models and connections.
- **Reliability:** Centralized data storage minimizes inconsistencies and ensures data integrity.
- **Security:** Controlled access to data protects sensitive information and facilitates audit trails.

- Performance optimization: Efficient data access and manipulation enhance app performance.
- Data-driven decision-making: Comprehensive data insights empower informed business choices.

Suitability for CUET Food Delivery Service

- Large data volume: Customer information, restaurant data, orders, deliveries, and financial transactions require efficient storage and management.
- Dynamic environment: Frequent changes in orders, menus, and delivery schedules necessitate a flexible architecture.
- Real-time interaction: Tracking order progress, location updates, and delivery confirmations requires fast data access.
- Business analytics: Data-driven insights are crucial for optimizing delivery routes, pricing strategies, and marketing campaigns.

10.3.2 UML Communication Diagram

Advantages

- Visualization: Clearly depicts interactions and data flow between system components.
- Communication: Facilitates efficient communication and collaboration among development teams.
- Documentation: Serves as a valuable documentation artifact for future reference and maintenance.
- Problem identification: Helps identify potential communication bottlenecks and synchronization issues.

Suitability for CUET Food Delivery Service

- Multiple stakeholders: Illustrates the interaction between customers, restaurants, delivery drivers, and the food delivery app.
- Order life-cycle: Clearly depicts the various stages of an order, from placement to delivery.
- Data flow: Visualizes the flow of data between different components, aiding in understanding the system's operation.
- Error identification: Helps identify potential communication errors or inconsistencies in the order process.

10.4 Technology, Software & Hardware

Certainly, here are concise descriptions for each of the mentioned languages in the context of the "CUET Food Delivery Service" mobile app:

- **Dart:** Dart is a programming language primarily used for building mobile, web, and desktop applications. It's the primary language for Flutter, a popular framework used in developing cross-platform mobile apps, including the CUET Food Delivery Service app.
- **C++:** C++ is a powerful and versatile programming language commonly utilized for performance-critical tasks in software development. While not typically used for mobile app frontend development, it might be employed in backend systems or for optimizing specific functionalities of the app.
- **CMake:** CMake is a build automation tool used for managing the build process of software across various platforms. It might be used in the development pipeline of the app to handle the compilation and building of components written in C++ or other languages.
- **Ruby:** Ruby is a dynamic, object-oriented programming language known for its simplicity and productivity. Although not commonly used for mobile app development itself, it could be employed in backend systems, scripting tasks, or web-related functionalities of the CUET Food Delivery Service.
- **Swift:** Swift is Apple's programming language primarily used for developing native iOS and macOS applications. It might be employed in the CUET Food Delivery Service app specifically for iOS development, enabling efficient and seamless functionalities for iOS devices.

These languages each serve distinct purposes within the development ecosystem, contributing to different aspects of the CUET Food Delivery Service app, from frontend UI design to backend functionality, cross-platform compatibility, and system optimization.

11. GUI (Graphical User Interface) Design

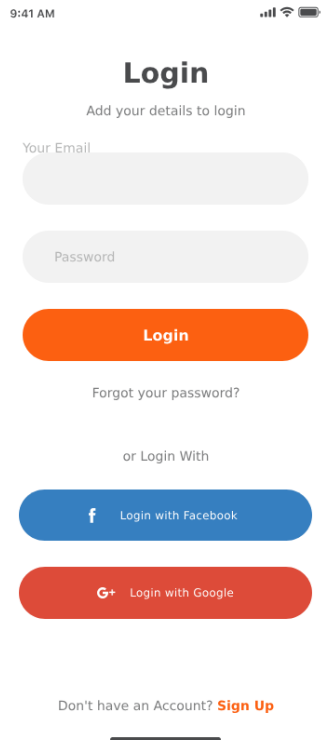
A graphical user interface (GUI) is a user interface that uses icons, menus, and other visual indications or representations to allow users to interact with electronic devices such as computers and smartphones (graphics). Simply described, a graphical user interface (GUI) is a technique of communicating with a computer program (or operating system) by utilizing graphical symbols rather than entering commands. GUIs allow us to communicate with a computer using picture-like elements (such as icons and arrows). To draw the user interfaces, we took help from Photopea [10] and Behance [11].



Figure 11.1: Splash



Figure 11.2: Login or Creat Account



9:41 AM

Login

Add your details to login

Your Email

Password

Login

Forgot your password?

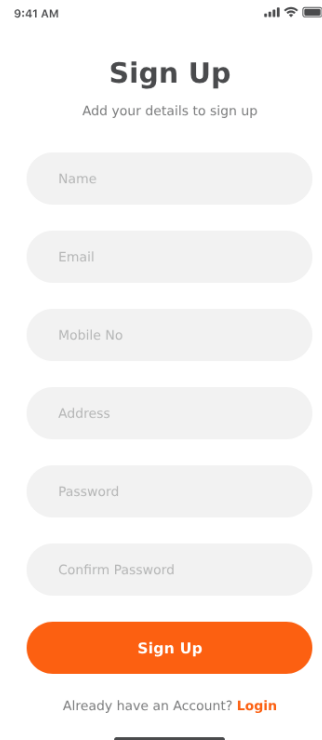
or Login With

Login with Facebook

Login with Google

Don't have an Account? [Sign Up](#)

Figure 11.3: Login Page



9:41 AM

Sign Up

Add your details to sign up

Name

Email

Mobile No

Address

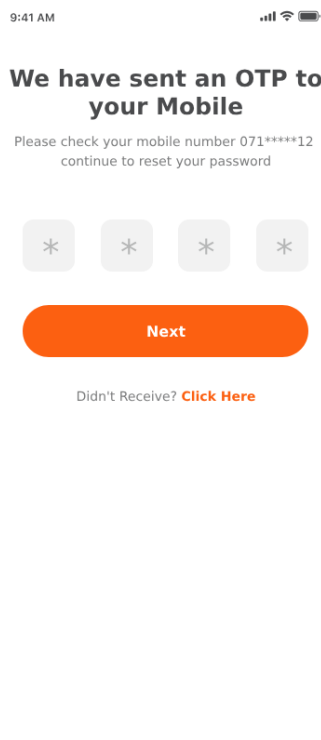
Password

Confirm Password

Sign Up

Already have an Account? [Login](#)

Figure 11.4: Sign Up



9:41 AM

We have sent an OTP to your Mobile

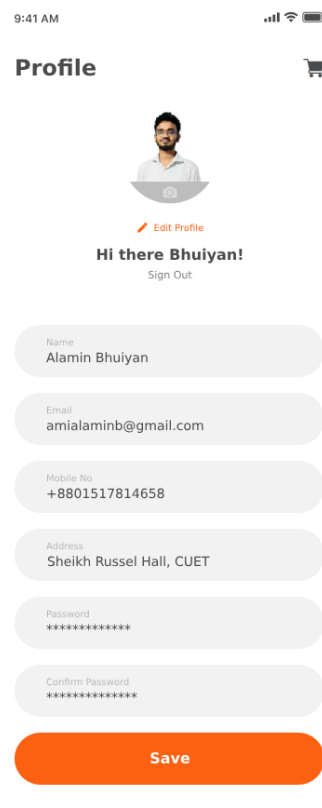
Please check your mobile number 071****12 continue to reset your password

* * * *

Next


Didn't Receive? [Click Here](#)

Figure 11.5: Confirm Mobile Number using OTP



9:41 AM

Profile



[Edit Profile](#)

Hi there Bhuiyan!

[Sign Out](#)

Name
Alamin Bhuiyan

Email
amialaminb@gmail.com

Mobile No
+8801517814658

Address
Sheikh Russel Hall, CUET

Password

Confirm Password

Save

Figure 11.6: User Profile

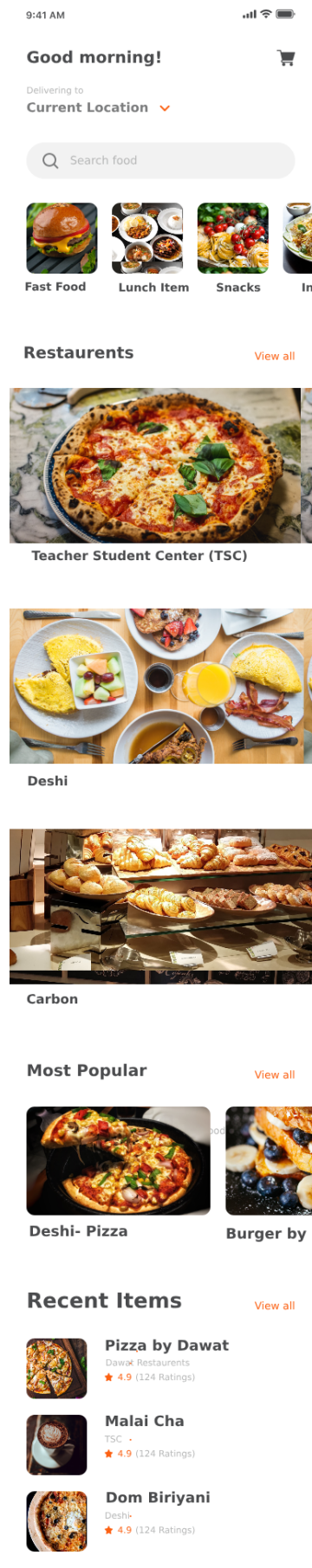


Figure 11.7: Home Page

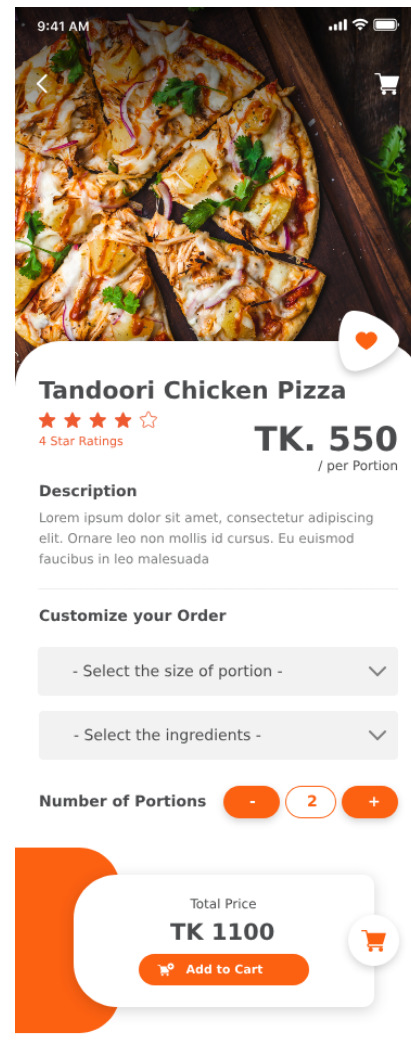


Figure 11.8: Food Details

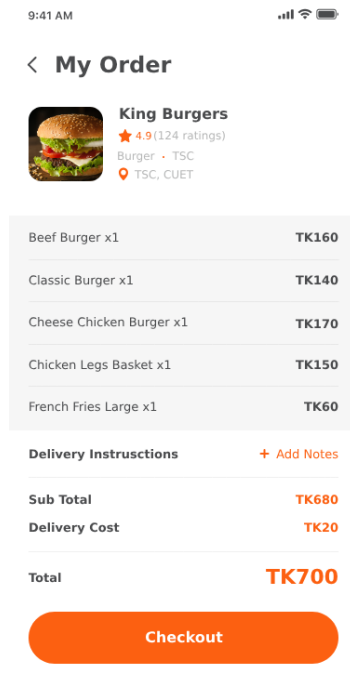


Figure 11.9: Order Details

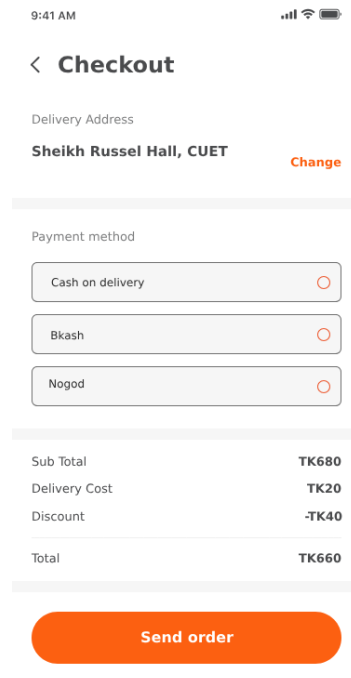


Figure 11.10: Checkout Information

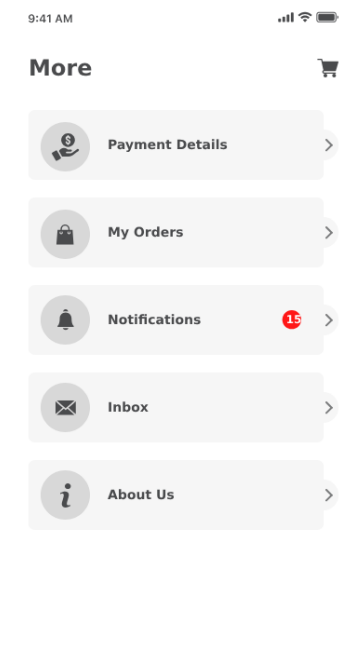


Figure 11.11: Order History and Other Informations

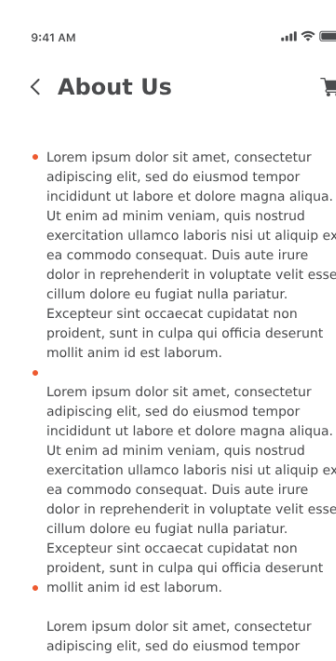


Figure 11.12: About Us

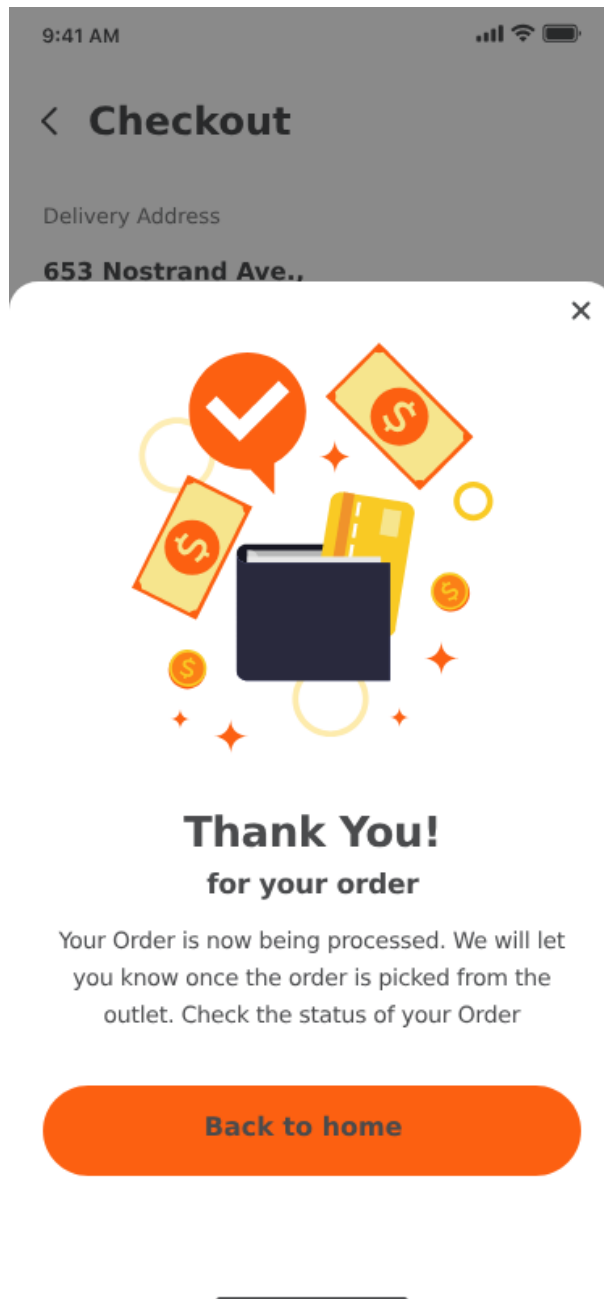


Figure 11.13: Order Confirmation Message

12. Design

12.1 Component level design pattern

The component-level design for the online food delivery app involves breaking down the system into smaller, self-contained components, each responsible for specific functionalities within the User Module, Order Food Module, and Food Management Module. In the User Module, the User Authentication Component manages login and registration, the User Profile Component handles user information and preferences, the Notification Component manages communication channels, and the Feedback Component deals with user reviews and ratings. Within the Order Food Module, the Menu Component retrieves and displays restaurant menus, the Cart Component manages the shopping cart and calculates order totals, the Order Processing Component handles order submission and communication with restaurants, the Payment Component manages secure transactions, and the Order History Component keeps track of user order history. For the Food Management Module, the Restaurant Management Component oversees restaurant profiles and menu items, the Order Fulfillment Component manages incoming orders and coordinates with the kitchen, the Inventory Management Component tracks ingredient availability, the Promotions Component handles restaurant promotions and discounts, and the Analytics Component provides insights into restaurant performance. These component-level designs aim to enhance modularity, scalability, and maintainability, ensuring that each component has well-defined responsibilities and interfaces, contributing to a more organized and efficient development process for the online food delivery app.

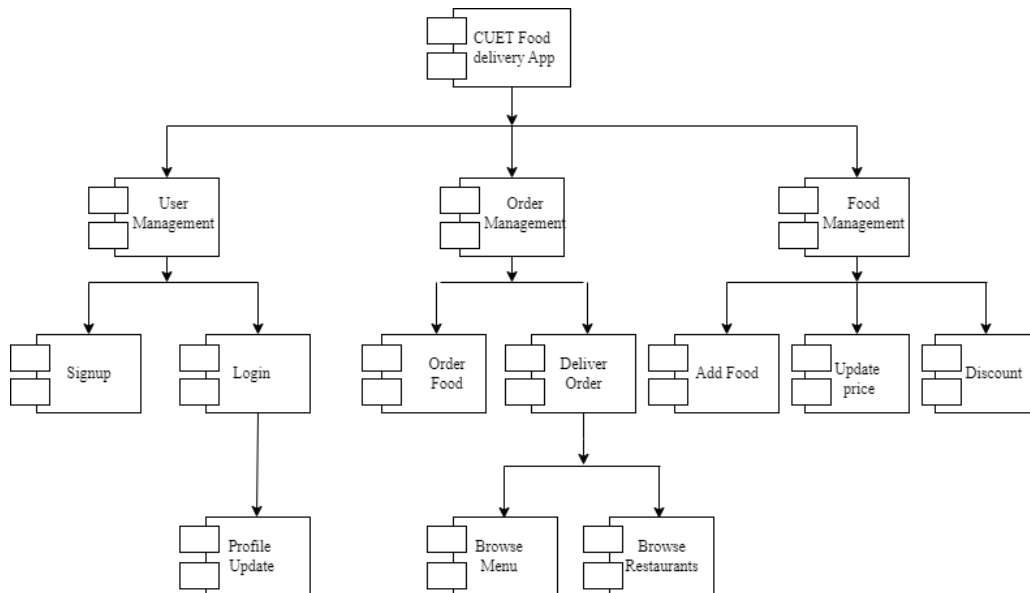


Figure 12.1: Structure Chart for CUET food delivery app

12.1.1 User Management

The User Management component within the CUET food delivery app serves as the central system responsible for handling all user-related functionalities. It encompasses modules that control user registration, authentication, profile management, and access

control. By dividing tasks into distinct components, it ensures efficient and secure handling of user data while offering a seamless experience to app users.

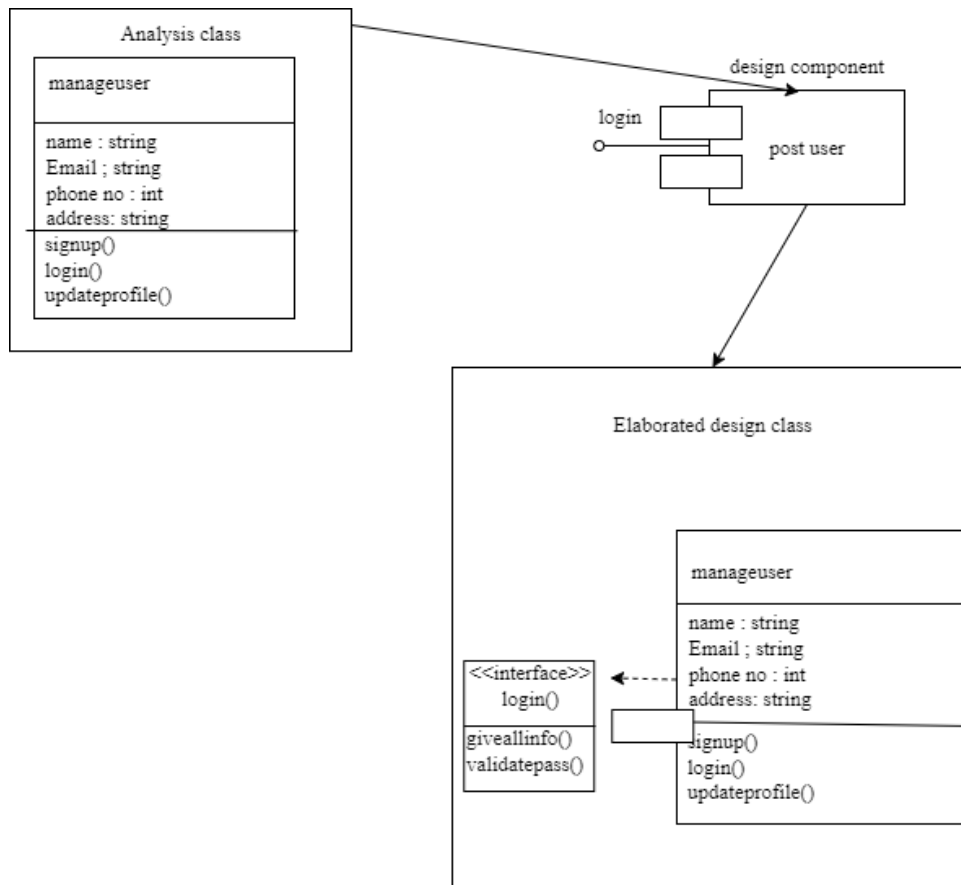


Figure 12.2: Component level design for user management.

12.1.2 Order Placement

The Order Placement component in the CUET food delivery app is responsible for facilitating the seamless and efficient placement of orders by users. It encompasses functionalities that allow users to browse menus, add items to their carts, customize orders, and finalize purchases. This component plays a pivotal role in ensuring a smooth and enjoyable user experience while ordering food through the app.

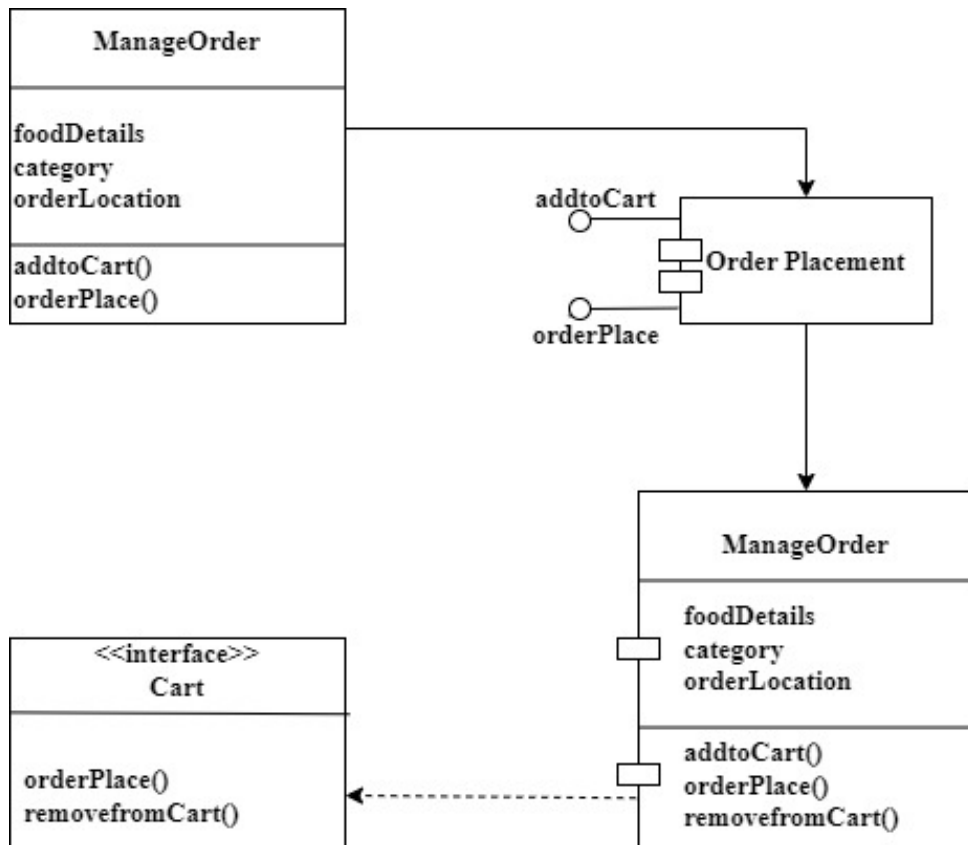


Figure 12.3: Component level design for Order Management.

12.1.3 Food Management

The Food Management component plays a pivotal role in managing the diverse range of food items offered within the CUET food delivery app. It encompasses functionalities related to food item cataloging, updating, and maintenance to ensure an up-to-date and appealing selection for users.

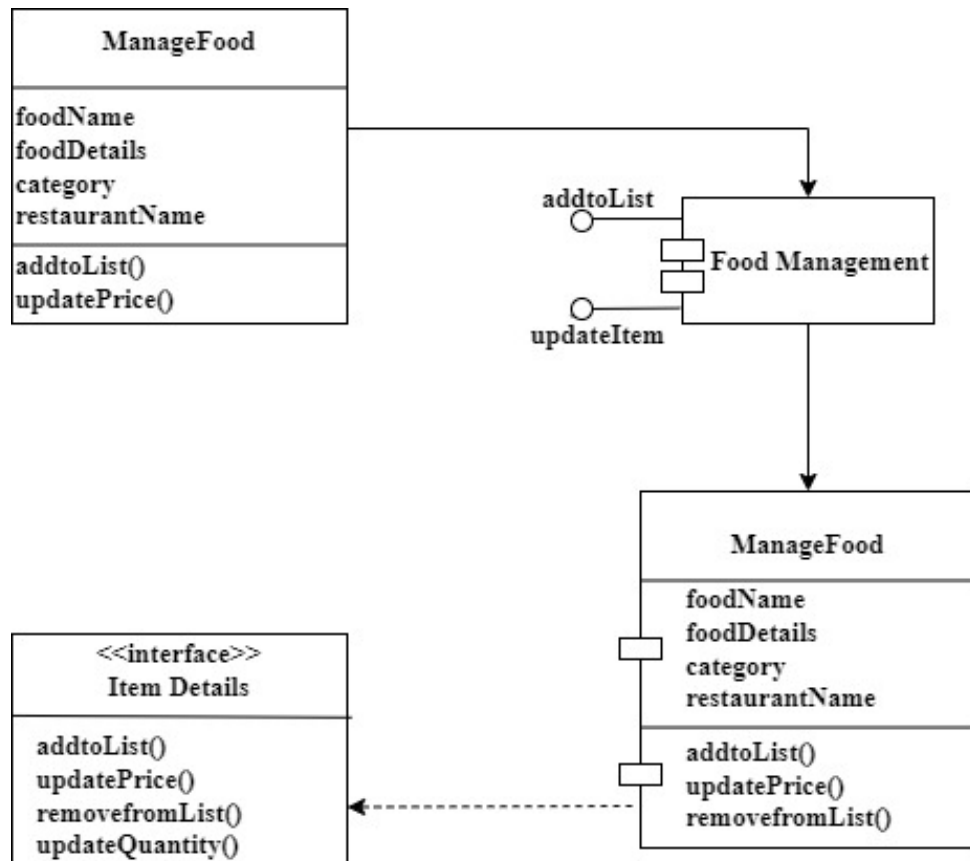


Figure 12.4: Component level design for user food management.

12.2 Dataflow Diagram (DFD)

12.2.1 First Level DFD for the System

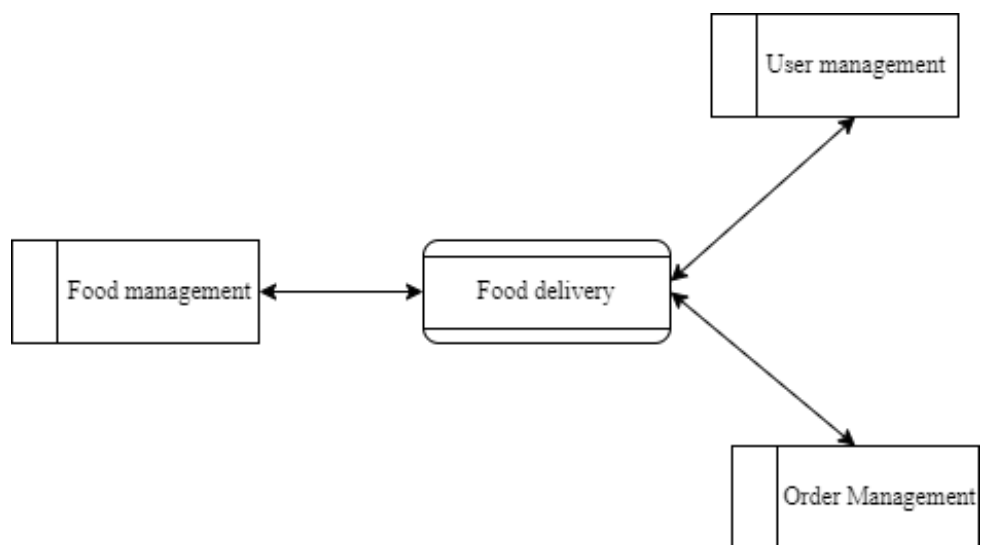


Figure 12.5: First Level Dataflow diagram for CUET Food delivery app

- User Module: Handles user authentication, profile management, notifications, and feedback.
- Food Management Module: Includes components for restaurant management, order fulfillment, inventory management, promotions, and analytics.
- Order Food Module: Manages menu display, shopping cart functionality, order processing, payment, and order history.

This 1st level DFD provides an overview of the major modules and their interactions within the online food delivery app system. It serves as a foundation for more detailed DFDs and system documentation.

12.2.2 DFD for user management

The User Management Data Flow Diagram (DFD) delineates the integral processes and data flows within the User Authentication, User Profile, Notification, and Feedback components of the online food delivery app system. It orchestrates user authentication through login and registration, manages user profiles, facilitates communication through notifications, and gathers and displays user feedback and ratings. The collaboration between components ensures a secure and seamless user experience, where authentication details, profile information, and feedback data flow seamlessly between components and the underlying user database. Together, these components form a robust user management system, crucial for the functioning and user satisfaction of the overall online food delivery application.

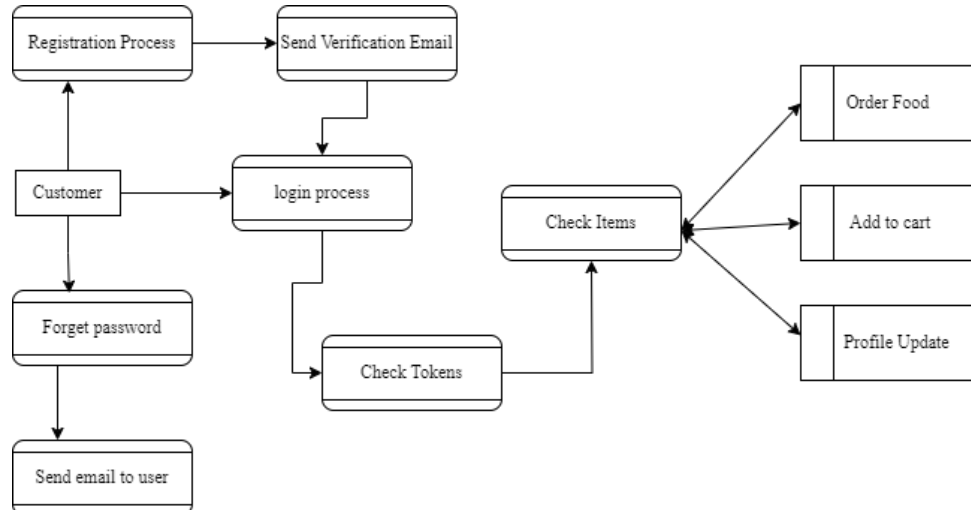


Figure 12.6: Dataflow diagram for user management

12.2.3 DFD for order management

The Order Management module of the Online Food Delivery System facilitates user interaction, order processing, payment handling, and order fulfillment. Users select items from the menu, manage their orders in the cart, and proceed to payment. The system processes orders, communicates with the restaurant for confirmation, securely handles payments, coordinates order fulfillment, and updates users through notifications. Completed orders are stored in the Order History for user reference. External systems, such as

payment gateways, are integrated for secure transactions. The module ensures a seamless and efficient workflow from order placement to delivery.

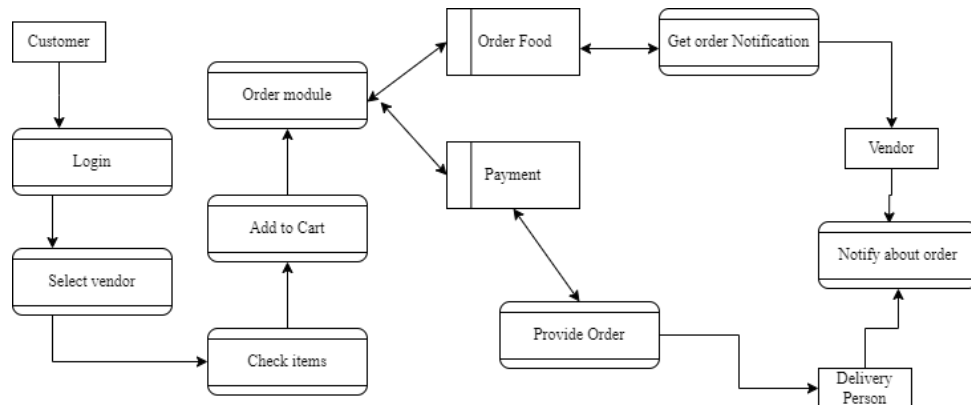


Figure 12.7: Dataflow diagram for order Management.

12.2.4 DFD for User Food Management

The Food Management Module of the online food delivery app comprises several key components. The Restaurant Management Component allows restaurants to register and edit profiles, as well as manage their menus. The Order Fulfillment Component is responsible for processing incoming orders and coordinating with the kitchen for order preparation. The Inventory Management Component tracks ingredient availability, issuing restocking notifications as needed, and manages stock levels. The Promotions Component empowers restaurants to create and manage promotional offers, displaying them to users. Lastly, the Analytics Component provides valuable insights to restaurants, including data on popular items and peak hours, aiding in strategic decision-making for optimal business performance. Together, these components contribute to effective restaurant management, streamlined order fulfillment, inventory control, promotional activities, and data-driven decision-making within the online food delivery app.

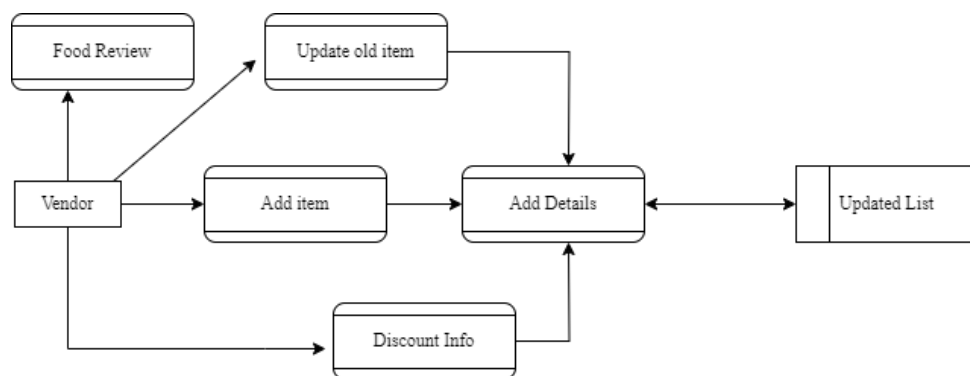


Figure 12.8: Dataflow diagram for Food management

12.3 Entity Relationship Diagram (ERD)

The Entity-Relationship (ER) diagram for the comprehensive online food delivery system encapsulates the fundamental entities and relationships pivotal to the system's function-

ality. Firstly, the 'User' entity represents the individuals utilizing the platform, engaging in activities such as placing orders, offering feedback, and managing personal profiles. The 'Restaurant' entity signifies the registered eateries on the platform, overseeing profiles, menus, and promotional activities. The 'Order' entity captures crucial information pertaining to user orders, encompassing details like items, quantities, and order statuses.

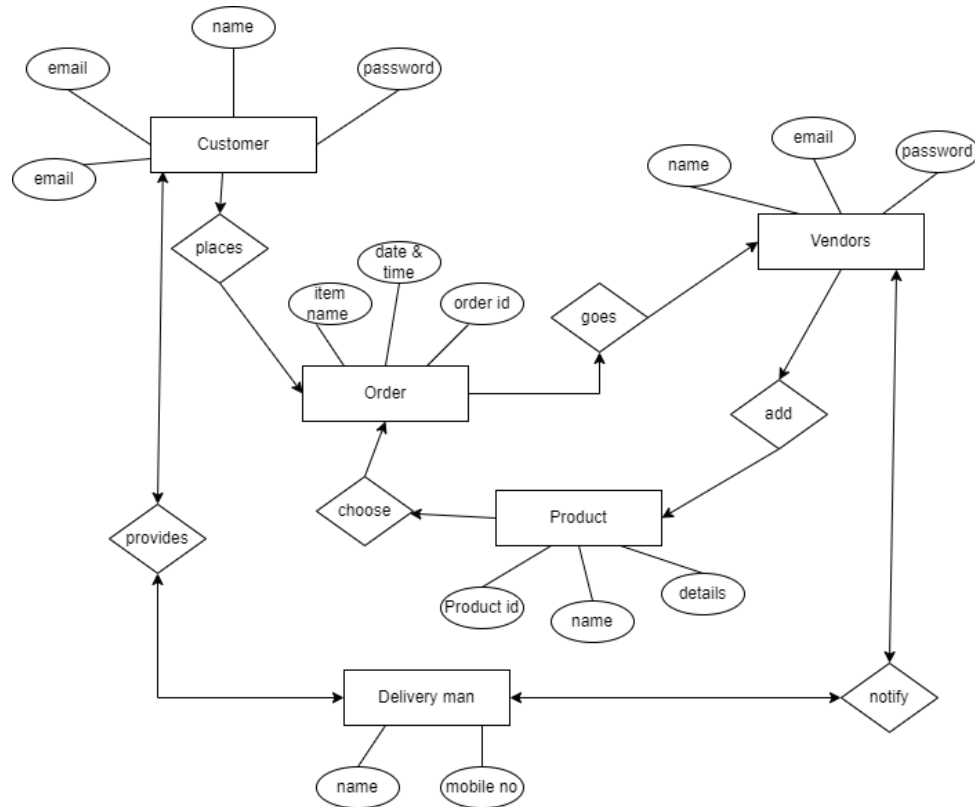


Figure 12.9: ER Diagram for food delivery system

13. API Documentation Manual

13.1 User Management

The documentation for the User Management API offers thorough information on the endpoints for adding, removing, updating, and retrieving users. It describes error handling, authentication techniques, and request and answer formats, making it easier to manage user-related actions within applications and integrate them seamlessly.

13.1.1 List Users

Request

Parameter	Values	Conditions	Expectations	Example
Email	String	Mandatory	Unique	alamin@gmail.com
Name	String	Mandatory	Name of user	alamin
Password	String	Mandatory	Password for login	123@ala
Contact Number	String	Mandatory	Contact Number of the user	01517814658
Address	String	Mandatory	Current Address of the user	Sheikh Russel Hall

Table 13.1: User Information Table



Figure 13.1: Header and Request URL

Response

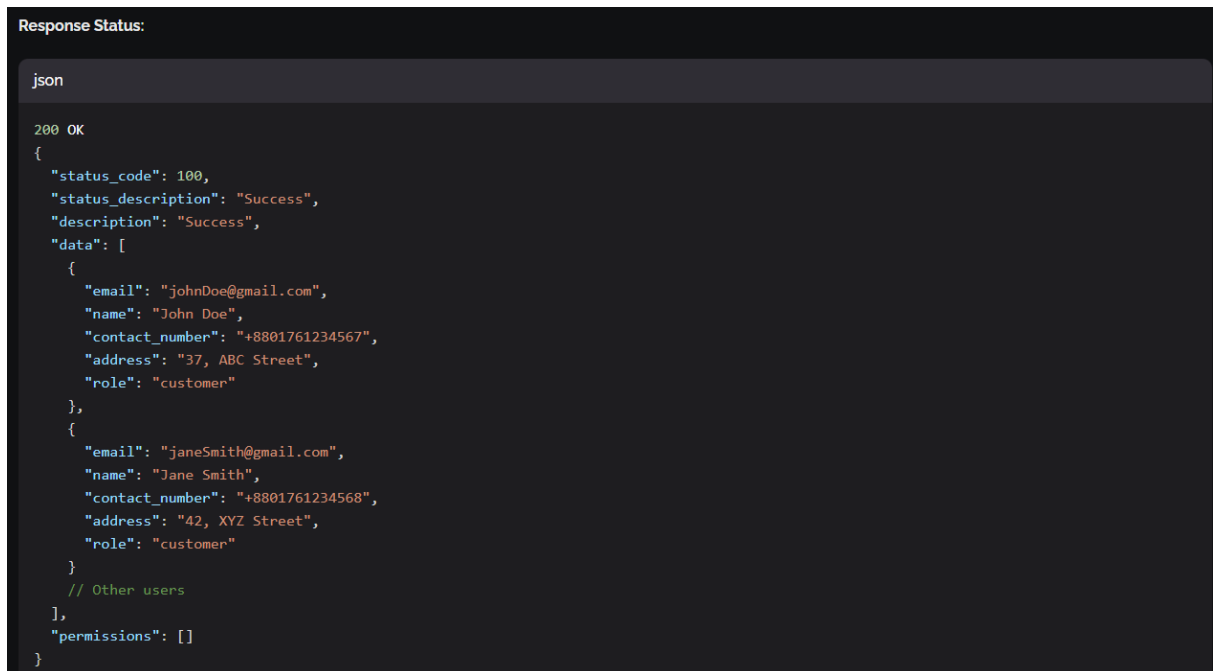


Figure 13.2: Response Body and response header

13.1.2 Add User

Request



Figure 13.3: Add User



Figure 13.4: Response of Add USER

Response

13.1.3 Update

Request

```
Request:

http

PUT /admin/users/4/update
Content-Type: application/json
Client-id: {{token}}
```

Figure 13.5: INPUT Parameters for the request JSON.

```
Response Status:

json

200 OK
{
  "status_code": 100,
  "status_description": "Success",
  "description": "User updated successfully",
  "data": {
    "email": "johnDoe@gmail.com",
    "name": "John Doe",
    "contact_number": "+8801670000000"
  }
}
```

Figure 13.6: Header and Request URL

14. Testing and Sustainability Plan

14.1 Requirements/Specifications-based System Level Test Cases

In the context of a Food Delivery App, the system-level testing process relies on a comprehensive specification document that outlines the requirements and functionalities of the application. The testing approach involves isolating each component for thorough examination to ensure alignment with end-user expectations and to affirm the system's reliability. A concise tabulated representation of system-level test cases, organized by requirements, facilitates systematic validation. The table below provides a simplified example:


Requirement ID	Requirement Statement 	Must/ Want	Comment
R-Registration-01	Users must be able to register an account on the app	Must	N/A
R-Registration-02	Display an error message if the user enters an invalid email during registration	Want	N/A
R-Login-01	Users must be able to log in with valid credentials	Must	N/A
R-Login-02	Display an error message for invalid login credentials	Want	N/A
R-Menu-01	Users should be able to browse the restaurant menu	Must	N/A
R-Menu-02	Display an indicator if an item is currently unavailable	Want	N/A
R-Order-01	Display a confirmation message after placing an order	Want	N/A
R-Feedback-01	Users can submit feedback and ratings for delivered orders	Must	N/A

Figure 14.1: Requirements/Specifications-based System Level Test Cases (CUET Food Delivery app)

15. Process Model

15.1 Process Model used

We employ the prototype process model paradigm to swiftly iterate and refine the "Cuet Food Delivery" app. Quick planning, iterative development, and stakeholder feedback guide the dynamic creation of a user-centric prototype.

- **Communication:** In the initial phase, engage with stakeholders, including potential users, restaurant partners, and delivery personnel. Define the overarching objectives for the "Cuet Food Delivery" app, such as delivering a user-friendly experience, streamlining order processing, and ensuring timely deliveries.
- **Quick plan:** The "Quick Plan" phase in the prototyping paradigm involves rapid planning and initiation activities to set the groundwork for the development process of CUET food delivery app. Such as, Stakeholder Communication, Identifying the primary objectives and priorities
- **Quick Design:** We develop a rapid design phase concentrating on components of the app visible to end users. Design the user interface with a focus on providing a smooth and intuitive experience for customers, restaurant owners, and delivery drivers. Consider the visual presentation of elements like order confirmations, delivery status updates, and payment receipts.
- **Construction of Prototype:** We completed the initial prototype of the "Cuet Food Delivery" app based on the swift design and implement essential functionalities, including menu browsing, order placement, and payment processing.
- **Deployment and Feedback:** We gather feedback from various stakeholders, encompassing users, restaurant partners, and delivery personnel. Utilize the gathered feedback to enhance and refine the prototype, making adjustments to align with user preferences and requirements. Conduct iterative deployments of new versions of the prototype, incorporating improvements and potentially introducing additional features.

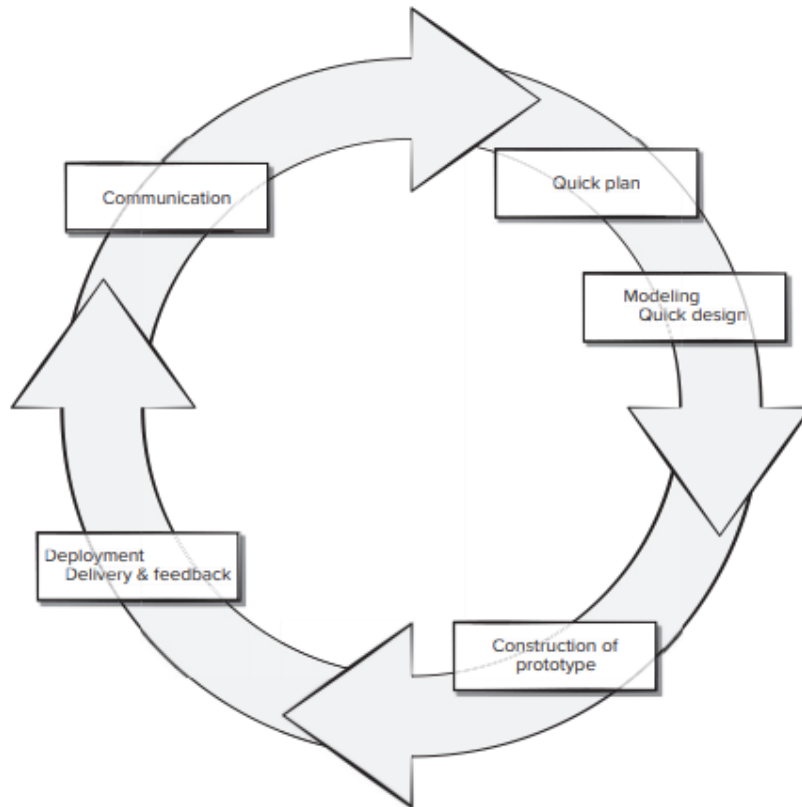


Figure 15.1: Prototyping Process Model

By adhering to the prototype process model, the development of the "Cuet Food Delivery" app can be dynamic, responsive to evolving requirements, and closely aligned with the needs and expectations of its diverse user base.

15.2 Rationale

Our choice for the software development process model for the "Cuet Food Delivery" app is the Prototype Process Model. Here's why we've opted for this model:

- **Iterative and Incremental Development:** The CUET food delivery app is dynamic, and requirements may evolve. The iterative and incremental nature of prototyping allows us to adapt and refine the app based on changing needs.
- **User-Centric Design:** We prioritize user experience in our app. The prototyping approach involves potential users early on, ensuring that the app's interface is intuitive and user-friendly.
- **Feedback and Stakeholder Involvement:** Prototyping encourages regular feedback from stakeholders, including users, restaurants, and delivery personnel. So we using prototype process model.
- **Flexible to Changes:** The food industry is dynamic, and market trends can change. The prototype process model accommodates changes effectively, allowing us to adjust based on market trends, user feedback, and emerging technologies.

- **Continuous Improvement:** The iterative nature of prototyping promotes continuous improvement. As the app evolves through multiple iterations, we can incorporate enhancements, new features, and optimizations based on lessons learned.
- **Risk Mitigation:** By obtaining early user feedback and addressing potential issues in the prototyping phase, we can mitigate risks associated with misunderstandings, unclear requirements, or changes in stakeholders' expectations.

In summary, we've chosen the Prototype Process Model for "Cuet Food Delivery" due to its adaptability, focus on user experience, and the ability to involve stakeholders in the development process from the early stages. This model aligns well with the dynamic and user-centric nature of developing a food delivery application.

16. Risk Analysis

16.1 Risk identification

Risk identification is a systematic attempt to specify threats to the project plan (estimates, schedule, resource loading, etc.). By identifying known and predictable risks, the project manager takes a first step toward avoiding them when possible and controlling them when necessary.

- **Product Size:** Inadequate scalability of the app to handle a large number of simultaneous users during peak hours. Overestimation or underestimation of the required features, leading to scope changes.
- **Business Impact:** Changes in local regulations affecting the operations of the food delivery service. Market competition leading to pricing pressures or customer acquisition challenges.
- **Stakeholder Characteristics:** Limited engagement from end-users in providing feedback on the app's usability.
- **Process Definition:** Deviations from planned development methodologies affecting the overall project timeline.
- **Development Environment:** Incompatibility issues with various mobile devices and operating systems.
- **Technology to be Built:** Complexity in integrating different restaurant menus and handling diverse cuisines.

16.2 Risk projection

Risk projection, also called risk estimation, attempts to rate each risk in two ways, the likelihood or probability that the risk is real and will occur and the consequences of the problems associated with the risk, should it occur. So, Risk projection for "Cuet Food Delivery App" involves assessing the likelihood and consequences of identified risks. By establishing scales, delineating consequences, estimating impacts, and assessing accuracy, the process prioritizes risks for effective resource allocation, ensuring a focused and efficient risk management strategy throughout the project's lifecycle.

16.3 Risk Refinement

In the context of the "Cuet Food Delivery App," risk refinement involves evolving broad risks into detailed scenarios for better mitigation. Using the Condition-Transition-Consequence (CTC) format, an initial risk related to reusable software components may be refined: Given that all reusable components must adhere to design standards, and some don't, there's concern that only 70% of planned modules may integrate. Refinement includes subconditions: third-party components lack internal design standards awareness and design standards for interfaces are undecided, and 30% custom engineering need. Despite

Risk	Category	Probability (%)	Impact
Size may be small	PS	50	2
Larger number of users than planned	PS	60	1
Technology will not meet expectations	TR	10	3
Lack of experience	ST	30	2
Less users than planned	PS	60	3
Customer will change requirements	PS	80	2
Lack of maintenance	BU	60	2

Table 16.1: Developing a Risk Table

concerns about the number of identified risks, Doug emphasizes considering all possibilities and plans to prioritize them later. This iterative process ensures comprehensive risk coverage and effective management for the "Cuet Food Delivery App."

16.4 Risk Mitigation, Monitoring and Management

16.4.1 Risk Mitigation

- Scalability Concerns: Conducting thorough performance testing and optimize the app for scalability.
- Data Security: Employing robust encryption protocols, conduct regular security audits, and ensure compliance with data protection regulations.
- Regulatory Compliance: Staying informed about local regulations, collaborate with legal experts, and integrate features that facilitate adherence to food safety and business regulations.
- Technological Dependencies: Diversifying technology dependencies, choose reliable third-party APIs, and establish contingency plans for service disruptions.

16.4.2 Risk Monitoring

- Regularly review and update the risk register to ensure it reflects the current project landscape.
- Implementation real-time monitoring tools to track app performance, user feedback, and potential security threats.

16.4.3 Risk Management

- Prioritization: Categorize risks based on their likelihood and impact, focusing on those with the highest potential consequences.

- Response Planning: Develop detailed response plans for high-priority risks, outlining specific actions, responsibilities, and timelines.
- Continuous Improvement: Conduct regular retrospectives to evaluate the effectiveness of risk management strategies and adjust them as needed.

17. Constraints to project implementation

17.1 Schedule

- Time Frame: We gotta figure out a realistic timeline for our project – you know, taking into account development, testing, and deployment and watching out for external factors messing with our schedule, like holidays or availability and study of our other courses.
- Milestones: We break down the project into milestones, each with its deadlines. We make sure these milestones fit into our overall project timeline.

17.2 Software & Hardware Constraints

- :Flutter and Dart Versions: We have to ensure compatibility with the targeted versions of Flutter and Dart. Regularly update dependencies to benefit from the latest features, improvements, and bug fixes.
- Third-Party Libraries: We evaluated the compatibility and stability of third-party packages used in the project. Some packages have constraints based on Flutter or Dart versions.
- Cross-Platform Compatibility: We verify that the app's features and UI elements are compatible across android platforms only.
- API Compatibility: We ensure compatibility with the targeted API levels for Android. This includes testing the app on different devices and screen sizes.
- Database Compatibility: We were choosing a database solution that is compatible with Flutter and Dart and verify that data storage and retrieval functions as expected.

18. Hardware and Software Resource (Tools/Language) Requirements

18.1 Software Requirements

- Development Environment: We use latest version of the Flutter SDK and Dart language and Integrated Development Environment (IDE) such as Visual Studio Code or Android Studio for Flutter development. Node.js runtime environment. Express.js framework for building the backend API is used.
- Version Control: We use git for version control. Platforms like GitHub or GitLab can be used for collaboration.
- Database: We choose a suitable database system based on project requirements.
- API Documentation: We use tools like Swagger for creating and testing API documentation.
- Package Management: Flutter relies on Dart's package manager, and Node.js uses npm. Ensure both Flutter and Node.js dependencies are managed efficiently.
- Testing Frameworks: Flutter provides its testing tools for UI and unit testing.

18.2 Hardware Requirements

- Development Machines: We have moderate performance machines with semi-sufficient RAM and processing power for smooth development. and SSD storage is recommended for faster build and compilation times.
- Testing Devices: Emulators is used for virtual testing during development.

19. Project Timeline and Schedule

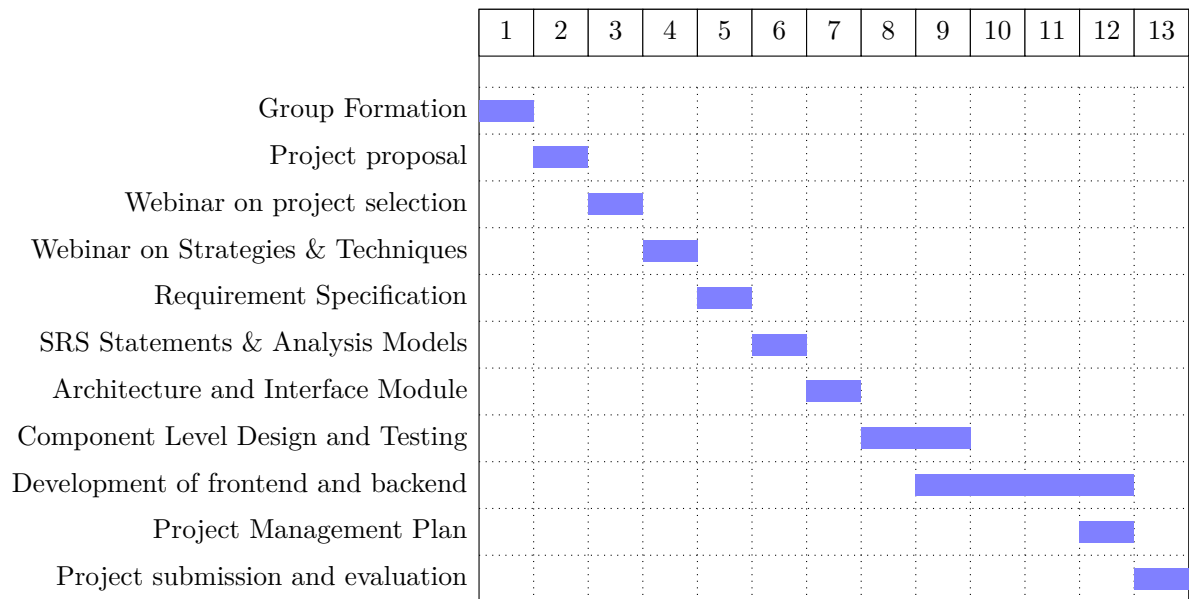


Figure 19.1: Project Gantt Chart

20. Estimated Budget

The estimated budget for the CUET Food Delivery Mobile App will be a thorough evaluation, taking into account various factors to ensure the successful development and deployment of the application within the specified scope and requirements. Numerous elements contribute to estimating the budget for the CUET Food Delivery Mobile App:

20.1 Type of Software Project:

- The CUET Food Delivery Mobile App is classified as a new mobile application development project.
- The nature of the project involves creating an intuitive and efficient mobile app for facilitating food delivery services, implying a significant development effort.

20.2 Size of Software Project:

- The project size is categorized as medium, considering the scope and complexity associated with developing a comprehensive food delivery application.
- As a medium-sized project, it includes various features such as user account management, menu browsing, order placement, payment processing, and real-time order tracking.

20.3 Development Team Size:

- The team size for the CUET Food Delivery Mobile App will be determined based on the specific requirements of the project.
- Key roles such as Project Manager, Mobile App Developers, UI/UX Designers, and QA Testers are essential for the successful development and deployment of the mobile application.
- The team size will be adjusted based on the workload and requirements of each role, ensuring a balanced and efficient development process.

20.4 Other Considerations:

- Additional factors, including the choice of technology stack, mobile platform considerations (Android), infrastructure requirements, will contribute to the overall budget estimation.

Taking into account these factors, a comprehensive budget estimation will be developed to cover all aspects of the CUET Food Delivery Mobile App development, ensuring a successful and robust application tailored to the specific needs of the Chittagong University of Engineering and Technology (CUET) community.

Table 20.1: Estimated Budget Table

Area of Cost	Cost (in TK)
Laptop x 3	2,40,000
Resource Cost - Other Software	2,000
Server	5,000
UI/UX Designer	10,000
Developer Cost - Developer x 3	60,000
Documentation and Others	3000
Total	3,20,000

21. Social, Cultural Environmental impact

21.1 Social impact

- **Convenience and Accessibility:** The app enhances convenience for users by providing easy access to a variety of food options. It caters to diverse dietary preferences and ensures that individuals, including those with busy schedules, can conveniently access meals.
- **Employment Opportunities:** The app creates job opportunities for delivery drivers and restaurant staff, contributing to economic growth and providing income sources for individuals in the food service industry.
- **Health and Well-being:** The app allows users to make informed choices about their meals, promoting healthier eating habits. It also encourages the consumption of diverse cuisines, contributing to cultural diversity.

21.2 Cultural impact

- **Culinary Diversity:** The app introduces users to a variety of cuisines, promoting cultural exchange and appreciation for different culinary traditions. It facilitates the exploration of diverse food options, enriching cultural experiences.
- **Preservation of Local Cuisine:** By supporting local restaurants, the app plays a role in preserving and promoting regional culinary traditions. It allows users to discover and enjoy authentic dishes from various cultures.

21.3 Environmental impact

- **Reduced Traffic and Emissions:** By offering food delivery services, the app may contribute to reducing individual trips to restaurants, leading to decreased traffic congestion and lower carbon emissions. Practices: Encouraging partner restaurants to adopt sustainable packaging and delivery practices can minimize the environmental footprint of food delivery services.
- **Waste Reduction:** Efforts to minimize packaging waste and promote eco-friendly practices within the app's ecosystem contribute to overall waste reduction, aligning with environmental sustainability goals.

In summary, the "Cuet Food Delivery App" has the potential to positively impact individuals by providing convenient access to diverse food options, supporting local economies, and promoting healthier eating habits. Additionally, it contributes to cultural exchange and integration while encouraging environmental sustainability through responsible business practices.

21. References

- [1] “Soft Robotics,” <https://www.softrobotics.com.bd/>.
- [2] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Use Case: Page-137.
- [3] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Activity Diagram: Page-147.
- [4] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Swimlane Diagram: Page-153.
- [5] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Class Diagram: Page-143.
- [6] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Sequence Diagram: Page-148.
- [7] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Architecture Style: Page-188.
- [8] “GeeksforGeeks,” <https://www.geeksforgeeks.org/software-engineering-architectural-design/>.
- [9] “Software Engineering A Practitioner’s Approach By Roger S. Pressman,” Architecture Style(UML): Page-190.
- [10] “Photopea,” <https://www.photopea.com/>.
- [11] “Behance,” <https://www.behance.net/onboarding>.