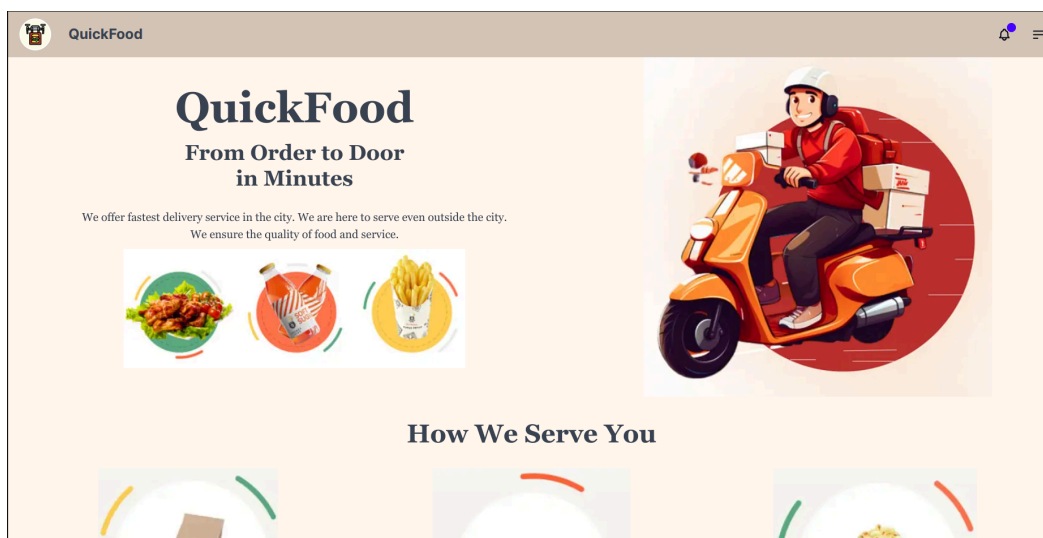


QuickFood: A Online Food Delivery Web App

Software Requirement Analysis Document (RAD)



1. Introduction:	3
1.1 Purpose of the system:	3
1.2 Scope of the System:	4
1.3 Objective and success criteria of the project:	4
1.4 Definitions and Acronyms and abbreviations:	5
1.5 References:	5
1.6 Overview:	6
2. Overall Description:	6
2.1 Project Perspective:	7
2.2 Product Functions Details:	7
2.3 User Profiles:	8
2.4 Constraints:	10
2.5 Assumptions and Dependencies:	11
3. Specific Requirements:	13
3.1 Overview:	13
3.2 Functional Requirements:	13
Requirement 1:	13
Requirement 2:	14
Requirement 3:	14
Requirement 4:	15
Requirement 5:	15
Requirement 6:	15
Requirement 7:	16
Requirement 8:	16
Requirement 9:	17
Requirement 10:	17
Requirement 11:	17
Requirement 12:	18
3.3 Non-Functional Requirements:	18
Usability:	18
Reliability:	18
Performance:	19
Supportability:	19
Implementation:	19
Scalability:	20
Security:	20
Testability:	20
Maintainability:	20
3.4 System Models:	21

3.4.1 Scenarios.....	21
1. Registers/Signs Up.....	21
2. Logs In.....	21
3. Edits Profile.....	22
4. Order Food.....	22
5. Cancel Order.....	22
6. Submit Review.....	22
7. Register Restaurants(for Restaurant Owner):.....	23
8. Handle Menu(for Restaurant Owner).....	23
9. Update Order Status(for Rider).....	24
10. Admin Privileges (for Admin).....	24
3.4.2 Use cases.....	24
Use case 1:.....	25
Use case 2:.....	25
Use case 3:.....	25
Use case 4:.....	26
Use case 5:.....	26
Use case 6:.....	27
Use case 7:.....	27
Use case 8:.....	28
Use case 9:.....	28
Use case 10:.....	28
Use case 11:.....	29
Use case 12:.....	29
Use case 13:.....	30
Use case 14:.....	30
3.4.3 Use Case Model.....	31
3.4.4 Dynamic model.....	32
Sequence Diagram.....	32
State Chart Diagram.....	33
1. User State Chart Diagram.....	33
2. Employee State Chart Diagram.....	34
Activity Diagram.....	35
3.4.5 Interfaces.....	36
3.4.5.1 User Interface.....	36
3.4.5.2 Software Interface.....	42
3.4.5.3 Hardware Interface.....	42
4 Supporting Information.....	43

1. Introduction:

1.1 Purpose of the system:

The purpose of the food delivery web app is to provide a convenient and efficient platform for users to order food from a variety of restaurants and have it delivered to their doorstep. The web app aims to simplify the process of ordering food by offering a user-friendly interface where customers can browse menus, place orders, make payments, and track delivery in real time.

Key benefits of the food delivery app include:

Convenience: Users can order food anytime, anywhere, without visiting restaurants or making phone calls.

Variety: The web app offers a wide selection of cuisines and restaurants, allowing users to explore different dining options.

Time-saving: Ordering through the web app saves time compared to traditional methods, as users can quickly browse menus, place orders, and track delivery status.

More business profit: The application allows restaurants to earn more by getting orders from remote people.

By providing a seamless and enjoyable food ordering experience, the food delivery web app aligns with the corporate goal of enhancing customer satisfaction and loyalty. Additionally, it contributes to the business strategy of expanding market reach and increasing revenue by tapping into the growing demand for food delivery services.

1.2 Scope of the System:

The software requirements delineated in this document are tailored for the QuickFood web application, version 1.0. This RAD encompasses the entirety of the QuickFood system, encompassing all its subsystems and functionalities. The document elucidates the comprehensive scope of the QuickFood platform, encompassing all necessary subsystems to support its operational facets. QuickFood endeavors to redefine the food delivery experience by offering a comprehensive solution that supersedes traditional platforms, integrating additional features such as seamless collaboration, advanced search and categorization capabilities, stringent authorization controls, and more, to enhance user satisfaction and operational efficiency.

1.3 Objective and success criteria of the project

The primary objective of QuickFood is to provide a seamless and convenient food delivery experience for users, facilitating easy access to a wide range of culinary options. Success for QuickFood will be measured based on several criteria:

1. **User Adoption:** QuickFood aims to attract a significant user base, with a focus on increasing active users and repeat usage.
2. **Efficiency:** The platform must ensure timely delivery of orders, with minimal delays and errors in processing.

3. **Customer Satisfaction:** QuickFood endeavors to garner positive feedback and ratings from users, reflecting high levels of satisfaction with the service, food quality, and overall experience.
4. **Business Growth:** The success of QuickFood will be evident through increased revenue streams, expanded market presence, and sustained profitability.
5. **Technological Stability:** The platform must maintain robustness and reliability, with minimal downtime and technical glitches, ensuring a smooth user experience.
6. **Expansion and Scalability:** Success will also be measured by QuickFood's ability to scale operations, expand into new markets, and accommodate a growing user base without compromising service quality.
7. **Competitive Positioning:** QuickFood aims to establish itself as a leading player in the food delivery industry, competing effectively with established rivals while offering unique features and value propositions to users.

Achieving these objectives will signify the success of the QuickFood project and its ability to fulfill the needs and expectations of both users and stakeholders alike.

1.4 Definitions and Acronyms and abbreviations

RAD = Requirement Analysis Document

UML = Unified Modeling Language

1.5 References

1. A. H. D. Bernd Brügge, Object Oriented Software Engineering Using UML, Patterns, And Java Third Edition. Prentice-Hall, Inc., 2010.

2. R. S. Pressman, Software Engineering A Practitioner's Approach Sixth Edition. McGraw Hill Higher Education, 2005.
3. Freecodecamp.org, Uml for beginners, <https://www.youtube.com/watch?v=WnMQ8HlmeXc> & ab_channel=freeCodeCamp.org, [Online; accessed 21-February-2023], 2022.
4. L. Chart, Uml with lucid chart, <https://lucid.app/>, [Online; accessed 21-February-2023], 2023.

1.6 Overview:

The QuickFood web application aims to revolutionize the food delivery landscape by providing a seamless and efficient platform for users to order their favorite meals from a variety of restaurants. With QuickFood, users can browse through an extensive selection of cuisines, place orders with ease, and enjoy doorstep delivery in a timely manner. The platform integrates advanced features such as real-time order tracking and secure payment options. QuickFood also prioritizes restaurant partnerships, ensuring a diverse array of dining options for users to explore. By leveraging cutting-edge technology and user-centric design, QuickFood strives to become the go-to destination for food delivery, offering convenience, choice, and satisfaction to customers while fostering growth and success for restaurant partners.

2. Overall Description:

2.1 Project Perspective:

The QuickFood web application represents a new, self-contained product within the realm of food delivery services. It is not a follow-on member of a product family nor a replacement for existing systems, but rather a standalone solution designed to offer an innovative and comprehensive food ordering and delivery experience. QuickFood is built from the ground up to address the evolving needs and preferences of modern consumers, providing a user-friendly interface, robust functionality, and seamless integration with external services such as payment gateways and restaurant partners. While QuickFood operates independently, it may interface with external systems such as GPS services for accurate delivery tracking and payment processors for secure transactions. This RAD focuses on defining the specific requirements and functionalities of the QuickFood web application, ensuring it delivers exceptional value and convenience to its users.

2.2 Product Functions Details

- **Restaurant and Food Discovery:** QuickFood will empower users to explore a diverse range of restaurants and culinary offerings through intuitive browsing and robust search functionalities. Users can discover nearby eateries, explore menus, and filter options based on cuisine, price range, and dietary preferences.
- **Order Management:** Users will have the ability to effortlessly place, customize, and cancel food orders according to their preferences. QuickFood will streamline the ordering process, allowing users to specify delivery instructions, select preferred delivery times, and manage multiple orders simultaneously.
- **Secure Payment Processing:** QuickFood will try to integrate reliable payment gateways to facilitate seamless and secure online transactions for food orders. Users can confidently complete payments using various methods such as credit/debit cards, digital wallets, or other preferred payment options.

- **Real-Time Order Tracking:** To enhance transparency and convenience, QuickFood will try to provide real-time tracking of order status and delivery progress. Users can monitor the journey of their orders from preparation to doorstep delivery, ensuring timely arrival and minimizing uncertainties.
- **User Account Management:** QuickFood will offer comprehensive user registration, login, and profile management features, enabling users to create and maintain personalized accounts. Users can update personal information, view order history, and save favorite restaurants for future orders.
- **Rating and Review System:** QuickFood will implement a robust rating and review system, allowing users to provide feedback on their food delivery experiences. Users can rate restaurants, delivery services, and individual food items, helping to foster transparency and accountability within the platform.
- **Restaurant Management Tools:** QuickFood will equip restaurant partners with dedicated management tools to streamline menu updates and order processing. Restaurants can efficiently manage incoming orders, update menu offerings, and track performance metrics to optimize operations.
- **Customer Support Services:** QuickFood will prioritize customer satisfaction by offering responsive customer support services. Users can access help resources, submit inquiries or complaints, and receive timely notification if needed.
- **Data Privacy and Security Measures:** QuickFood will uphold stringent data privacy and security standards to safeguard user information and transactional data. The platform will employ encryption protocols, secure authentication mechanisms, and regular security audits to protect user privacy and prevent unauthorized access or breaches.

2.3 User Profiles:

QuickFood web application caters to several distinct user classes, each with unique roles, needs, and privileges:

1. Food Delivery Riders:

- **Characteristics:** Food delivery riders are individuals responsible for transporting orders from restaurants to customers' locations. They typically possess strong navigational skills and the ability to work efficiently in fast-paced environments.
- **Functionality:** Riders utilize the QuickFood app to receive order assignments, view delivery instructions, and navigate optimal routes for timely deliveries. They will also provide updates on order status and communicate with customers as needed.

2. Restaurant Owners:

- **Characteristics:** Restaurant managers oversee the operations of their respective establishments, including menu management and order processing. They require access to administrative functionalities to efficiently manage their restaurant's presence on the QuickFood platform.
- **Functionality:** Restaurant managers utilize the QuickFood admin dashboard to update menu items, manage order queues, and monitor performance metrics. They may also interact with customer feedback and reviews to maintain quality standards and enhance customer satisfaction.

3. Customers:

- **Characteristics:** Customers are individuals who use the QuickFood app to browse restaurants, place food orders, and track delivery status. They may vary in preferences, dietary requirements, and ordering habits.
- **Functionality:** Customers access the QuickFood platform to explore restaurant listings, browse menus, place orders, and make payments securely. They can track the status of their orders in real-time and provide feedback on their dining experiences.

4. Admin:

- **Characteristics:** The admin user class possesses elevated privileges and responsibilities for overseeing the overall operation and management of the

QuickFood platform. Admins ensure smooth functioning, enforce policies, and handle escalated issues.

- **Functionality:** Admins have access to comprehensive administrative tools within the QuickFood platform to manage user accounts, resolve disputes, enforce policies, and perform system maintenance tasks. They play a crucial role in ensuring platform integrity, security, and adherence to regulatory standards.

2.4 Constraints:

QuickFood web application development is subject to several constraints that shape the implementation and deployment of the platform:

1. **Regulatory Compliance:** QuickFood must adhere to relevant regulations and legal requirements governing food delivery services, including health and safety standards, data protection laws, and payment processing regulations. Compliance with these regulations may impose constraints on data handling, user privacy, and financial transactions.
2. **Hardware and Infrastructure:** The performance and scalability of QuickFood may be influenced by hardware limitations, including server capacity, network bandwidth, and storage resources. Developers must optimize the application to operate efficiently within the constraints of available hardware infrastructure.
3. **Interfaces and Integrations:** QuickFood relies on seamless integration with external systems and services, such as payment gateways, mapping APIs for location services, and restaurant management tools. Developers must ensure compatibility and reliability when interfacing with these external platforms, considering potential constraints on data formats, communication protocols, and API limitations.

4. **Technology Stack:** The choice of technologies, frameworks, and databases used in QuickFood development is constrained by factors such as organizational preferences, existing infrastructure, and technical expertise. We used Next js and tailwind css in frontend and spring boot and MySQL in backend.
5. **Security Considerations:** QuickFood must implement robust security measures to protect user data, financial transactions, and platform integrity. Constraints related to security may include encryption standards, access control policies, secure coding practices, and vulnerability management protocols.
6. **Design and Development Standards:** QuickFood development may be governed by design conventions, programming standards, and coding guidelines established by the customer's organization or industry best practices. Developers must adhere to these standards to ensure consistency, reliability, and maintainability of the software solution.
7. **Operational Considerations:** QuickFood may have operational constraints related to system availability, performance targets, maintenance schedules, and disaster recovery plans. Developers must design the application to meet operational requirements while considering constraints on resource availability, downtime tolerance, and service level agreements.

2.5 Assumptions and Dependencies:

Assumptions:

1. **Minimal Hardware Requirements:** It is assumed that the device running the QuickFood program meets the minimal hardware specifications outlined for optimal performance.
2. **Reliable Internet Connection:** Users are expected to have access to a reliable internet connection to use the QuickFood program without interruptions.

3. **Stability of Backend Infrastructure:** The assumption is made that there will be no unexpected difficulties with the application's backend infrastructure, ensuring smooth development progress without significant delays or failures.
4. **Absence of Schedule Conflicts:** It is assumed that there will be no schedule conflicts or time constraints during the development process that could adversely impact the project's timeframe.
5. **Basic Computer Literacy:** Users are expected to possess basic computer literacy skills, enabling them to navigate and utilize QuickFood's features effectively.
6. **Clear Articulation of Questions:** The assumption is made that users will be able to accurately and clearly articulate their questions and issues, facilitating effective communication and problem resolution within the platform's community.

Dependencies:

1. **Next JS:** QuickFood relies on Next JS for building its user interface, leveraging its modern and efficient front-end framework capabilities.
2. **Tailwind css:** We will use tailwind css in the front-end for UI design.
3. **SpringBoot Java Framework:** The backend of QuickFood is built using the SpringBoot Java Framework, ensuring the development of a robust and reliable Rest API system.
4. **MySQL:** The app uses a MySQL database in the backend for data storage.
5. **PostMan API:** PostMan API is utilized for testing and debugging the Rest API of QuickFood, enhancing the quality and reliability of the backend functionality.
6. **Maven and NPM:** QuickFood utilizes Maven and NPM for efficient and flexible application building processes, ensuring streamlined management and deployment workflows.

3. Specific Requirements:

3.1 Overview:

The food delivery web app serves as a convenient solution catering to the needs of both food enthusiasts and restaurant proprietors. Offering a diverse selection of cuisines and eateries, the platform enables users to effortlessly browse menus, place orders, and track deliveries in real-time, enhancing their dining experience. Simultaneously, restaurant owners benefit from increased visibility and access to a broader customer base, allowing them to efficiently manage orders and streamline operations. Admin and other employees, including delivery riders, play pivotal roles in ensuring the smooth functioning of the app, facilitating seamless communication and coordination between users and restaurants. By bridging the gap between food lovers and eateries, the food delivery web app revolutionizes the way people dine, providing convenience and satisfaction for all parties involved.

3.2 Functional Requirements:

Requirement 1

ID: QuickFood001

Name: Registration

Description: The registration process allows new users to create accounts on the food delivery web app. Users need to provide essential information such as name, email address, and password. The system should validate user input to ensure accuracy and completeness. Upon successful registration, users should receive a confirmation message or email.

Priority: High

Reference: This requirement falls under the user management and onboarding functionality of the app. It aims to provide a user-friendly experience for new users, enabling them to access personalized features and services while ensuring data integrity and security.

Requirement 2

ID: QuickFood002

Name: Authentication

Description: Authentication is required for users to access restricted features and make transactions on the food delivery web app securely. Users can authenticate themselves through email/password whereas employees can do so using employee ID and password. The authentication process verifies the user's identity and grants access to authorized functionalities based on their role.

Priority: High

Reference: This requirement is fundamental for ensuring the security and integrity of the app. It aligns with the app's security protocols and user access control mechanisms, ensuring that only authenticated users can interact with the system and perform authorized actions.

Requirement 3

ID: QuickFood003

Name: Update profile

Description: The "Update Profile" functionality allows users to modify their personal information after registration. Users will be able to edit details such as name, phone number, address, and profile picture. The system should validate user input to ensure data accuracy and completeness.

Priority: High

Reference: This feature falls under the user account management functionality of the app, enabling users to maintain up-to-date information and tailor their preferences according to their needs.

Requirement 4

ID: QuickFood004

Name: View food items

Description: Users will be able to browse and view a list of available food items offered by restaurants. The list will include details such as the name of the dish, description, price, and possibly images to provide a visual representation. Users will also have the option to filter or search for specific food items based on categories, cuisines, or price.

Priority: High

Reference: This requirement is fundamental to the core functionality of the food delivery app, as it will enable users to explore the available food options before placing an order.

Requirement 5

ID: QuickFood005

Name: Place order

Description: Once users have selected the desired food items, they will be able to place an order seamlessly through the app.

Priority: High

Reference: This requirement aligns with the primary objective of the app, which is to facilitate food ordering and delivery services.

Requirement 6

ID: QuickFood006

Name: Payment

Description: After placing an order, users will proceed to confirm payment to complete the transaction. Users will enter payment details securely, and the app will validate the transaction before processing the order.

Priority: High

Reference: Secure and reliable payment processing is crucial for ensuring user trust and facilitating seamless transactions.

Requirement 7

ID: QuickFood007

Name: Cancel order

Description: Users will have the ability to cancel an order. If the cancellation is due to late delivery, the user will get a refund.

Priority: High

Reference: Offering the option to cancel orders reflects the app's commitment to customer satisfaction and flexibility.

Requirement 8

ID: QuickFood008

Name: Submit rating

Description: After receiving their order, users will be able to submit a review and rating for the restaurant and their food. The ability to submit ratings encourages user engagement and fosters a sense of community within the app.

Priority: High

Reference: This requirement facilitates transparency and accountability, allowing users to share their experiences.

Requirement 9

ID: QuickFood009

Name: Register and update restaurants

Description: Restaurant owners will have the capability to register their establishments on the platform and subsequently update their restaurant profiles as needed. During the registration process, owners will provide essential details such as the restaurant name, location, contact information etc. Additionally, owners can update their profiles to reflect changes in business information.

Priority: High

Reference: This requirement is vital for empowering restaurant owners to leverage the food delivery app as a platform to showcase their offerings and connect with potential customers.

Requirement 10

ID: QuickFood010

Name: Manage Menu

Description: Restaurant owners will have the ability to manage their menus within the app, including adding, editing, and removing food items. The menu management functionality is essential for enabling restaurants to showcase their culinary offerings effectively.

Priority: High

Requirement 11

ID: QuickFood011

Name: Rider interface

Description: Rider will be able to get notification of orders and update order status accordingly.

Priority: High

Reference: This requirement is necessary for delivering the order within time.

Requirement 12

ID: QuickFood012

Name: Administrative privilege

Description: Admin will be able to handle users, other employees, restaurants and overall management of the app.

Priority: High

3.3 Non-Functional Requirements:

Usability:

The software will feature a user-friendly interface with intuitive navigation and easily understandable controls to enhance user experience. Consistency in layout and design will be maintained throughout the application to minimize confusion and ensure ease of use. Additionally, efficient data entry and retrieval mechanisms will be implemented to streamline administrative tasks. A user-friendly search function will also be integrated to provide accurate and relevant results with minimal effort.

Reliability:

The software will be designed to ensure continuous availability, allowing users to access it at all times. Robust error recovery mechanisms will be in place to prevent data loss or database corruption in the event of errors or crashes. Furthermore, the software will be capable of accommodating concurrent users without experiencing performance

degradation. Regular updates and maintenance activities will be conducted to uphold stability and reliability standards.

Performance:

The system will prioritize responsiveness, aiming to respond to user inputs within 2-3 seconds or less to maintain a seamless user experience. Additionally, the system will be engineered to support a minimum of 100 concurrent users without any performance deterioration, ensuring optimal performance under varying loads.

Supportability:

Efforts will be made to ensure ease of installation and configuration, with comprehensive documentation and installation guides provided to assist users. The system will be modular and easily maintainable, facilitating clear separation of concerns between different components and modules. To minimize downtime, automatic backup and recovery options will be available. Furthermore, the system will be compatible with diverse hardware and software environments, with detailed guidelines and recommendations provided to users for seamless integration.

Implementation:

The software will be built using Java and JavaScript, leveraging their robust capabilities to develop a comprehensive solution. MySQL will be employed as the preferred database management system (DBMS) to ensure efficient data storage and retrieval. Version control software, such as Git, will be utilized throughout the development process to facilitate collaboration among team members and effectively manage code changes, ensuring seamless integration of updates and enhancements.

Scalability:

The system is expected to accommodate a minimum of 500-1000 concurrent users without experiencing significant performance degradation. Critical transactions should maintain a response time of 3-5 seconds, even during peak load periods. To achieve this, the database will be designed to handle large volumes of data efficiently, implementing indexing and partitioning strategies to optimize performance.

Security:

Security measures will be implemented to ensure a secure login process for all users and employees. Each login will require a unique identifier and password. Role-based access control will be enforced to restrict users' access to authorized functions and data only. The system will prioritize the privacy and confidentiality of user data, such as personal information and browsing history. Additionally, an audit trail will be maintained to track all user activities, aiding in the detection and investigation of security breaches.

Testability:

The software will be architected in a modular manner to facilitate isolated testing of individual components. Testing and debugging processes will be rigorously executed to ensure the highest level of reliability for our system.

Maintainability:

The product will feature a modular and well-structured design, emphasizing clear separation of concerns to enhance comprehension and ease of modification. Extensive documentation will accompany the codebase to streamline maintenance and updates.

Additionally, the product will be engineered with scalability in mind, enabling seamless integration of new features and functionalities with minimal disruption to existing code.

3.4 System Models:

3.4.1 Scenarios

1. Registers/Signs Up

- When the user hits the Register button, the user will be routed to the registration page.
- On the registration page, the user will enter his name, email address, password.
- After requesting registration with the necessary information, the user will get a response from our system.
- If the system responds with a successful message the user will be redirected to the home page.
- If the system responds with a message other than successful, the warning/error message will be shown to the user.

2. Logs In

- If the user isn't logged in, by clicking the log-in button, the user will be routed to the login page.
- User will provide email/employee_id and password to log in.
- System will authenticate and authorize the user.
- If the user is authenticated, the system will respond with the necessary data and the user will be redirected to the home page.
- If any error occurs, the system will show an error/warning message.

3. Edits Profile

- On the edit page, the user will be able to change the name, password and other information.
- If the user provides valid info, he can update his profile.

4. Order Food

- The user will first go to the order page.
- The user can apply filters on food items.
- The user will choose the desired food items and specify quantities.
- After finalizing the order, the user will proceed to the payment.
- The system will verify the order details and prompt the user to confirm the order.
- Upon successful confirmation, the system will process the order and provide an order confirmation message to the user.

5. Cancel Order

- If the user wishes to cancel an order, they will navigate to the order history or active orders section.
- The user will select the order they want to cancel and initiate the cancellation process.
- The system will verify the cancellation request and prompt the user for confirmation.
- Upon confirmation, the system will cancel the order and provide a cancellation confirmation message to the user.
- If the cancellation request cannot be fulfilled, the system will display an error message explaining the reason.

6. Submit Review

- After receiving the ordered items, the user can navigate to the order history section.

- The user will select the completed order for which he wants to submit a review.
- On the order details page, the user will find an option to submit a review or feedback.
- The user will provide a rating for the ordered items or overall experience.
- Upon submission, the system will process the rating and display a confirmation message to the user.
- If the review submission fails due to any reason, the system will prompt the user to try again or display an error message.

7. Register Restaurants(for Restaurant Owner):

- Restaurant owners will access the registration page specifically designed for them.
- On this page, they will input details such as restaurant name, location, contact information, and menu items.
- After providing the required information, restaurant owners will submit the registration request.
- The system will validate the information and create a new restaurant profile if the details are accurate.
- Upon successful registration, restaurant owners will receive a confirmation message and gain access to their restaurant dashboard.

8. Handle Menu(for Restaurant Owner)

- Restaurant owners will navigate to their dashboard where they can manage their menu items.
- They will have options to add new menu items, update existing ones, or remove items that are no longer available.
- Restaurant owners can specify details such as item name, description, price, and availability status.

- After making changes to the menu, restaurant owners will save the updates, and the system will reflect the changes accordingly.
- If any errors occur during the menu management process, the system will display relevant error messages for correction.

9. Update Order Status(for Rider)

- Riders will log in to their account and access the order management section specifically designed for them.
- In the order management interface, riders will see a list of assigned orders along with their current status.
- Riders can select an order from the list and update its status based on the delivery progress (e.g., picked up, delivered).
- After updating the order status, riders will save the changes, and the system will automatically update the order status for all relevant stakeholders.
- If there are any issues with updating the order status, the system will provide error notifications for resolution.

10. Admin Privileges (for Admin)

- Admin will have access to an administrative dashboard with advanced functionalities.
- In the admin dashboard, they can view and manage various aspects of the system, including user accounts, restaurant profiles, and employee data.
- Admins can perform tasks such as adding or removing employees, performance analysis etc.

3.4.2 Use cases

Use case 1:

Name : Sign up

Actor: User

Flow of events:

- User hits the sign up button.
- On the sign up page, the user will enter his name, email address, password.
- After requesting sign up with the necessary information, the user will get a response from our system.

Entry Condition: User doesn't have an account.

Exit Condition: Get confirmation message.

Use case 2:

Name : Log in

Actor: User, Employee

Flow of events:

- User or employee hits the login button.
- On the login page, the user will enter email address, password and employee will enter employee ID and password.
- System will authenticate and authorize the user.
- If the user is authenticated, the system will respond with the confirmation message.

Entry Condition: User doesn't have an account.

Exit Condition: Get confirmation message.

Use case 3:

Name : Edit profile

Actor: User, Employee

Flow of events:

- The user first goes to his account.
- On the edit page, the user will be able to change the name, password and other information.
- If the user provides valid info, he can update his profile.

Entry Condition: User must be logged in.

Exit Condition: Get confirmation message for profile update.

Use case 4:

Name : Order Food

Actor: User

Flow of events:

- The user will first go to the order page.
- The user can apply filters on food items.
- The user will choose the desired food items and specify quantities.
- The system will verify the order details and prompt the user to confirm the order.
- User confirms the order.

Entry Condition: User must be logged in.

Exit Condition: Order is created.

Use case 5:

Name : Confirm payment

Actor: User

Flow of events:

- The user proceeds to the payment page after placing an order.
- The user enters card details and authentication details.
- The user will be verified by the bank.

- For verified users, the payment will be complete.

Entry Condition: User must have a card.

Exit Condition: Payment is done.

Use case 6:

Name : Cancel order

Actor: User

Flow of events:

- The user will navigate to the order history.
- The user will select the order they want to cancel and initiate the cancellation process.
- The system will verify the cancellation request and prompt the user for confirmation.

Entry Condition: User must place an order first.

Exit Condition: Order is canceled.

Use case 7:

Name : Submit review

Actor: User

Flow of events:

- The user will select the completed order for which he wants to submit a review.
- The user will provide a rating for the ordered items or overall experience.
- Upon submission, the system will process the rating and display a confirmation message to the user.

Entry Condition: User must receive the order first.

Exit Condition: Rating is submitted.

Use case 8:

Name : Update order status

Actor: Rider

Flow of events:

- Riders will log in to their account and access the order management section specifically designed for them.
- Riders can select an order from the list and update its status based on the delivery progress (e.g., picked up, delivered).
- After updating the order status, riders will save the changes, and the system will automatically update the order status for all relevant stakeholders.

Entry Condition: Rider must login first.

Exit Condition: Get confirmation message of update status.

Use case 9:

Name : Register rider

Actor: Admin

Flow of events:

- The admin goes to the admin dashboard.
- The admin enters details of the rider.
- The system confirms the registration.

Entry Condition: .Admin must login first.

Exit Condition: Confirmation of registration.

Use case 10:

Name : Admin analytics

Actor: Admin

Flow of events:

- The admin proceeds to the admin dashboard.
- The admin can view and analyze the performance of the employees and restaurants via graphs and charts.

Entry Condition: User must be logged in.

Exit Condition: Data is accessed.

Use case 11:

Name : Register Restaurants

Actor: Restaurant owner

Flow of events:

- Restaurant owners will access the registration page specifically designed for them.
- On this page, they will input details such as restaurant name, location, contact information, and menu items.
- After providing the required information, restaurant owners will submit the registration request.
- The system will validate the information and create a new restaurant profile if the details are accurate.

Entry Condition: Restaurant owner must be logged in first.

Exit Condition: Registration is done.

Use case 12:

Name : Update Menu

Actor: Restaurant owner

Flow of events:

- Restaurant owners will navigate to their dashboard where they can manage their menu items.
- They will have options to add new menu items, update existing ones, or remove items that are no longer available.

- After making changes to the menu, restaurant owners will save the updates, and the system will reflect the changes accordingly.

Entry Condition: Restaurant owner must be logged in first.

Exit Condition: Update is done.

Use case 13:

Name : Restaurant analytics

Actor: Restaurant owner

Flow of events:

- The restaurant owner proceeds to the restaurant dashboard.
- The owner can view and analyze the sell rate and performance of the restaurants via graphs and charts.

Entry Condition: Restaurant owner must be logged in.

Exit Condition: Data is accessed.

Use case 14:

Name : Logout

Actor: User, Employee, Restaurant owner

Flow of events:

- The actor clicks on the logout button.
- System will confirm the logout and navigate to the home page.

Entry Condition: Actor must be logged in.

Exit Condition: Logged out successfully.

3.4.3 Use Case Model

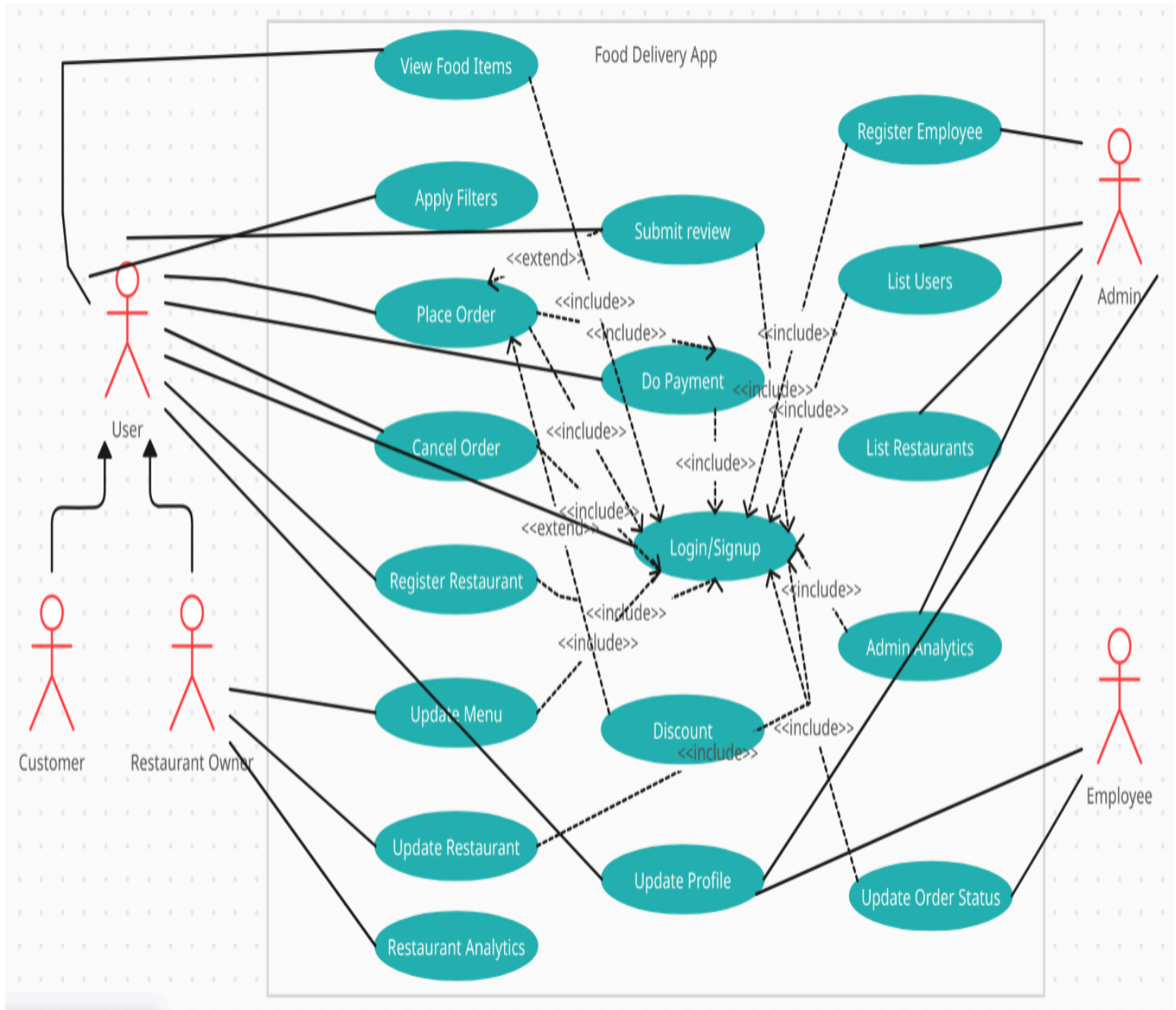


Fig : Use case model

3.4.4 Dynamic model

Sequence Diagram

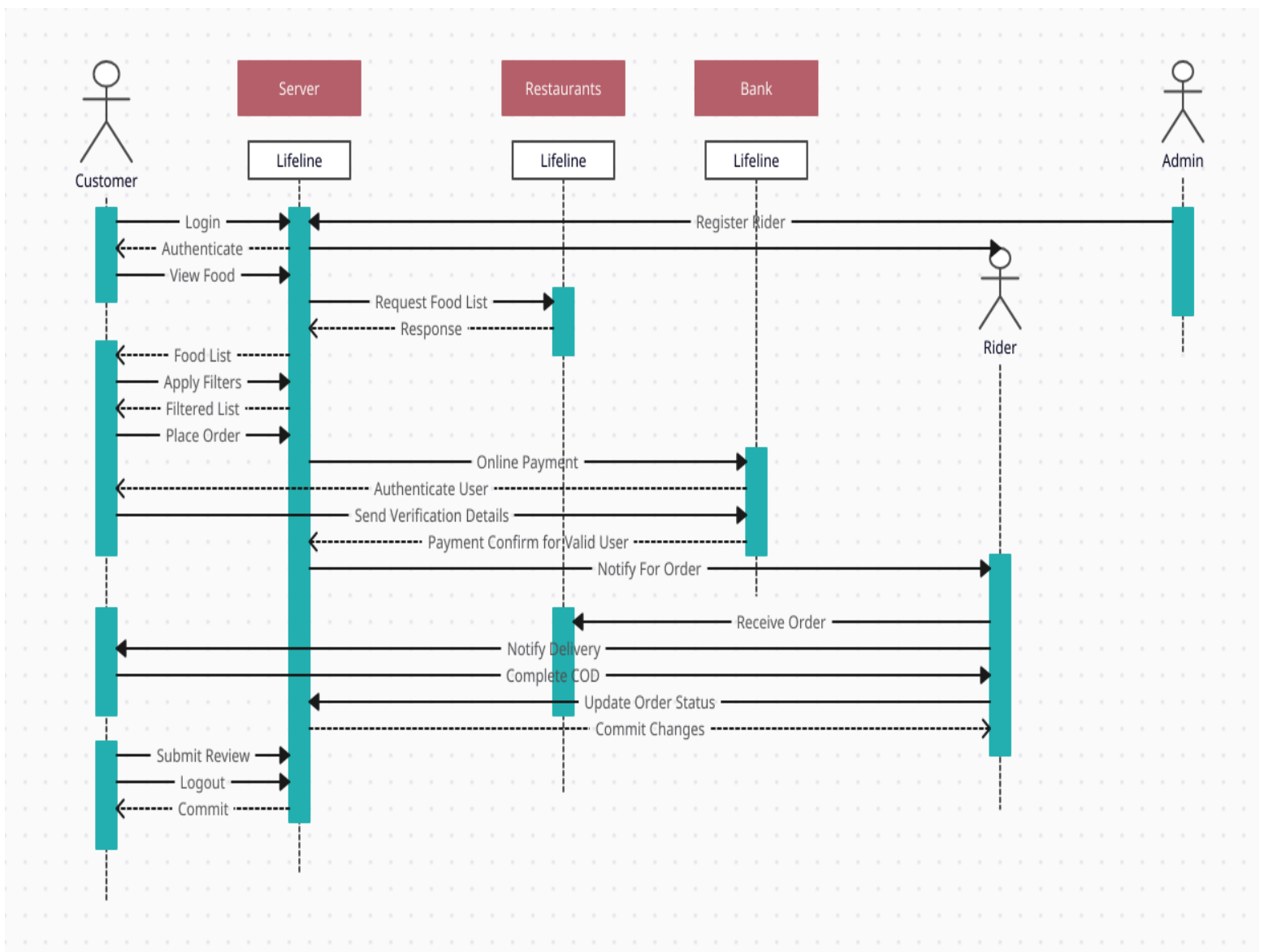


Fig : Sequence Diagram

State Chart Diagram

1. User State Chart Diagram

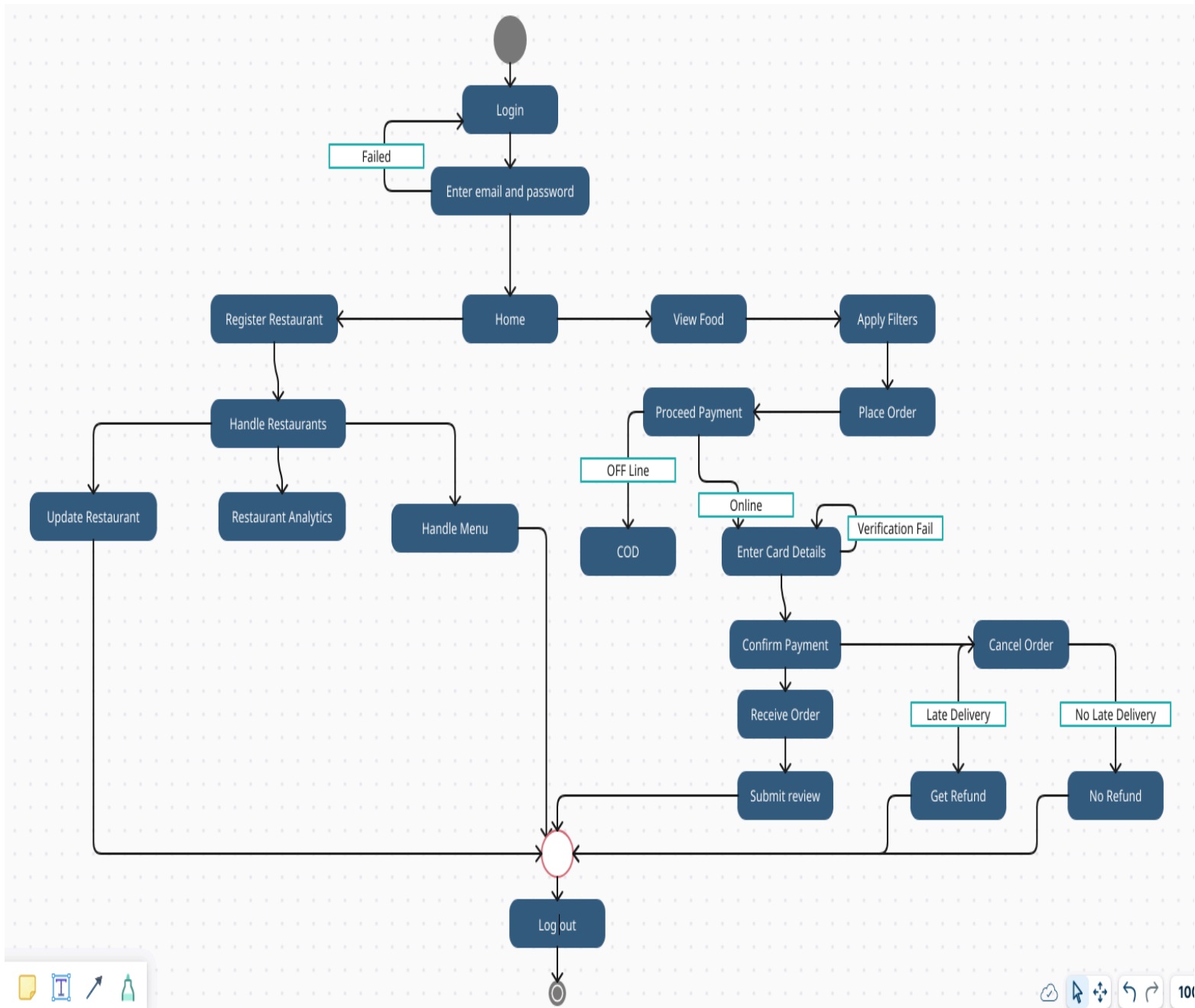


Fig : User State Chart Diagram

2. Employee State Chart Diagram

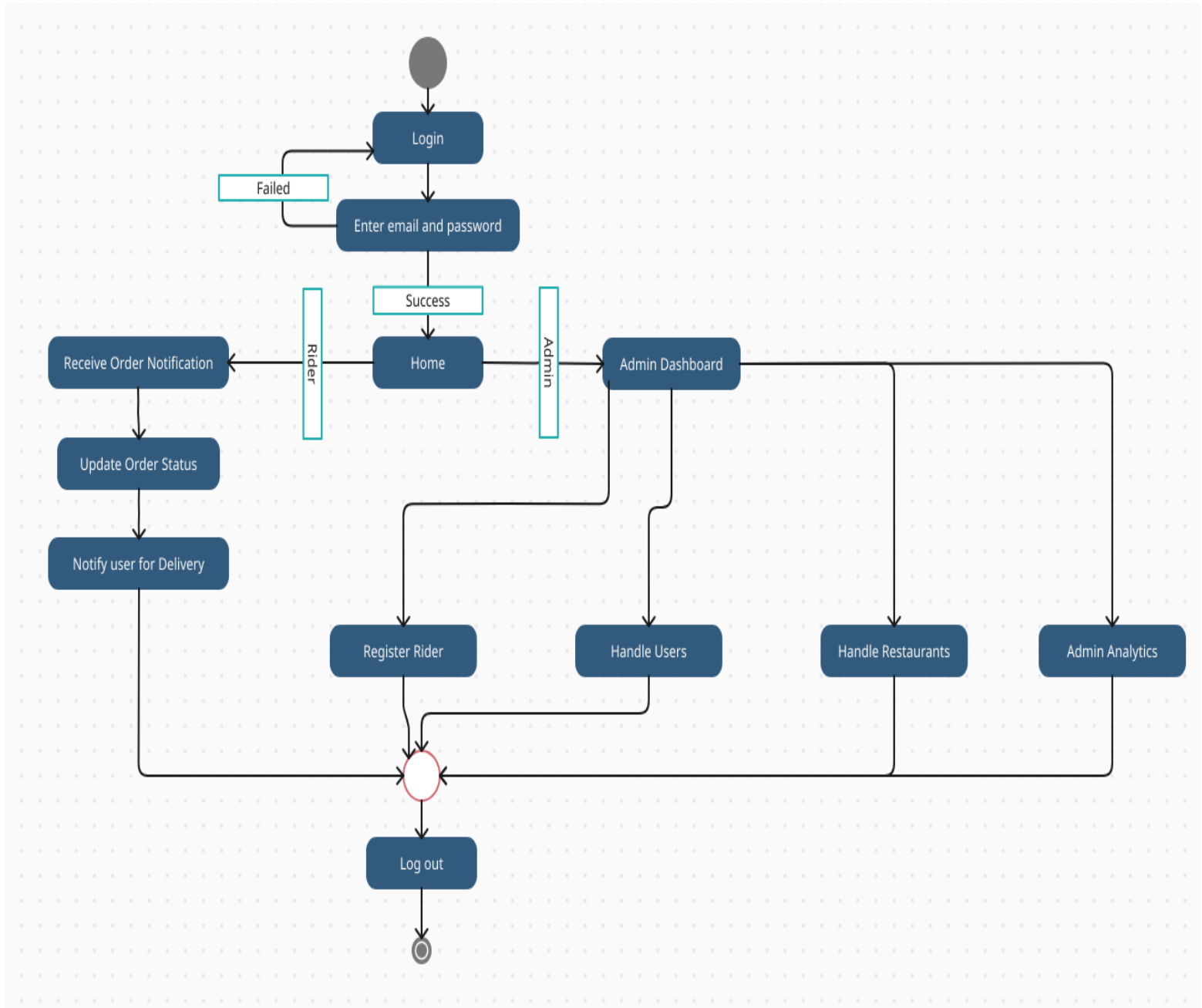


Fig : Employee State Chart Diagram

Activity Diagram

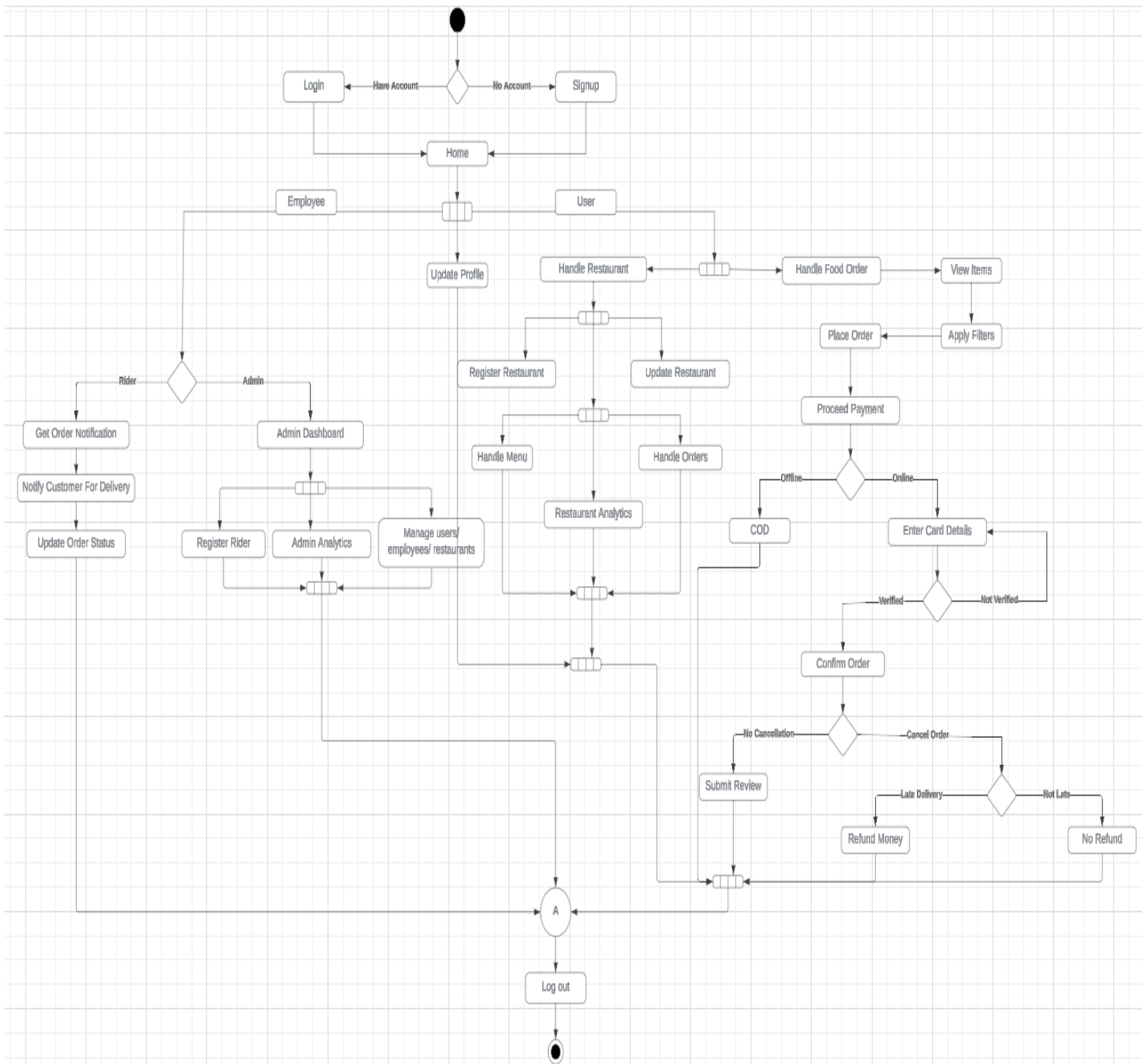


Fig : Activity Diagram

3.4.5 Interfaces

3.4.5.1 User Interface

The User Interface of “QuickFood” is designed according to conventional design standards with thoughtful implementations of different components. The User Interface includes the following pages.

Login: The login page provides users with a secure entry point to access the application. Users input their credentials to authenticate their identity and gain access to their accounts.

Sign Up: The sign-up page allows new users to create accounts within the application. Users provide necessary information such as username, email, password. After successful registration, users can log in with their newly created credentials.

Profile: The profile page displays user-specific information and settings. Users can view and edit their personal details, update contact information, manage delivery addresses, and adjust account settings such as notifications.

Home: The home page serves as the main landing page of the application, providing users with an overview of relevant content and features. It will contain buttons to navigate in other pages of the app.

Order Food: The order food page enables users to browse through available food items, explore menus from various restaurants, and place orders for delivery or pickup. Users can filter search results, view detailed item descriptions, select quantities, and confirm

payments. This page streamlines the ordering process, making it easy for users to find and order their favorite dishes.

Restaurants: The restaurants page showcases a directory of participating restaurants, allowing users to discover new dining options or explore familiar favorites. Users can browse through restaurant profiles, view menus, read reviews, check operating hours, and access location information. Restaurant owners can add or update their restaurant information and menu items. They can also monitor their performance and sell rate as well.

Order Status: The order status page provides users with real-time updates on the status of their orders. Users can track the progress of their orders from confirmation to preparation, dispatch, and delivery. The page displays relevant details such as order number, estimated delivery time, current status, and tracking information, keeping users informed and engaged throughout the delivery process.

Rider will update the order status accordingly.

Notifications: The notifications page delivers important updates and alerts to users regarding their order status. Users can receive notifications in real-time via push notifications and sockets.

Admin Dashboard: The admin dashboard serves as a centralized platform for administrators and staff to manage various aspects of the application. It provides access to administrative tools and features for monitoring user activity, updating menus, analyzing performance metrics, and configuring settings. The dashboard offers comprehensive insights and controls to streamline administrative tasks and optimize operational efficiency.

Some images of the UI

Home Page:

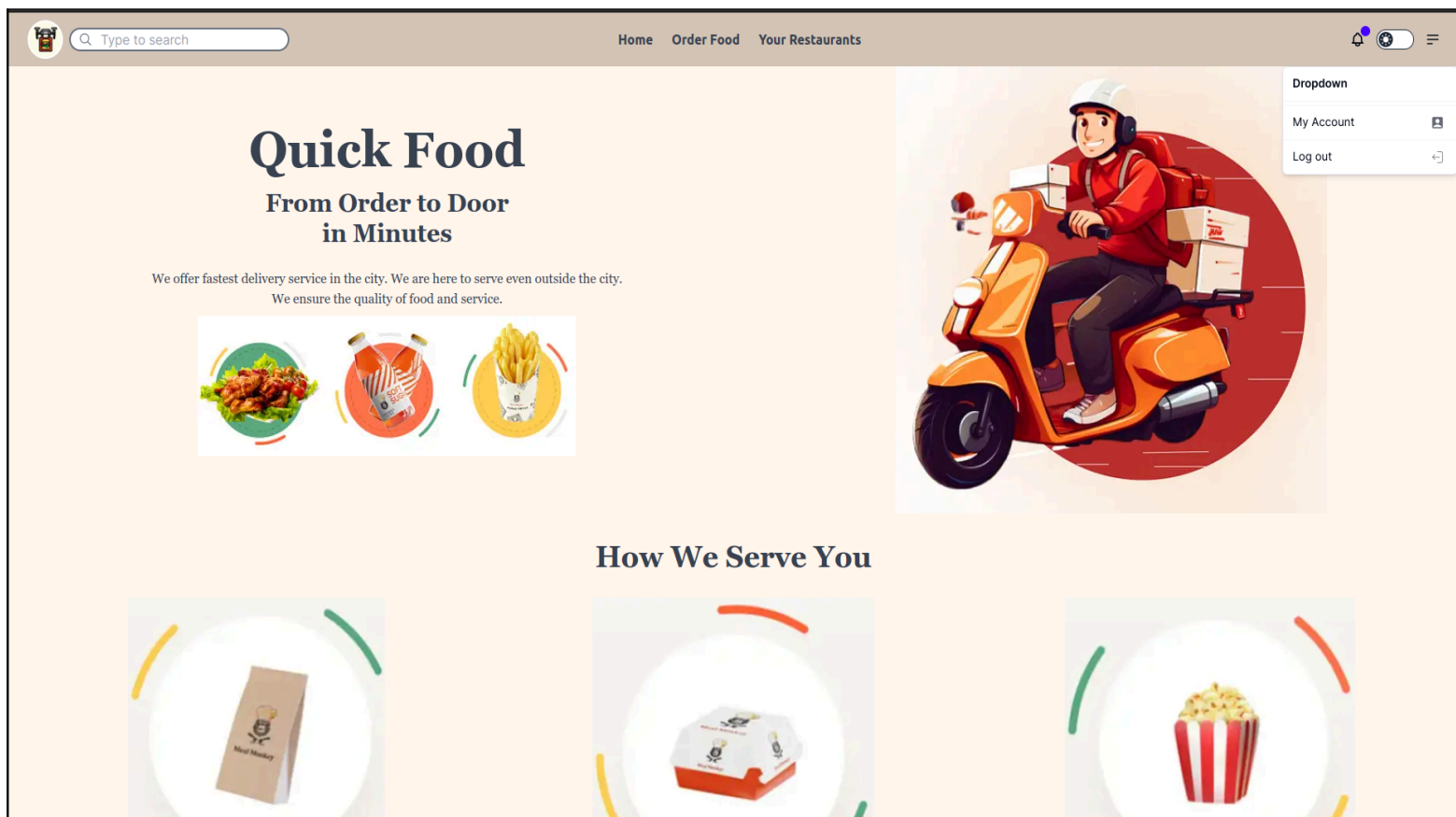
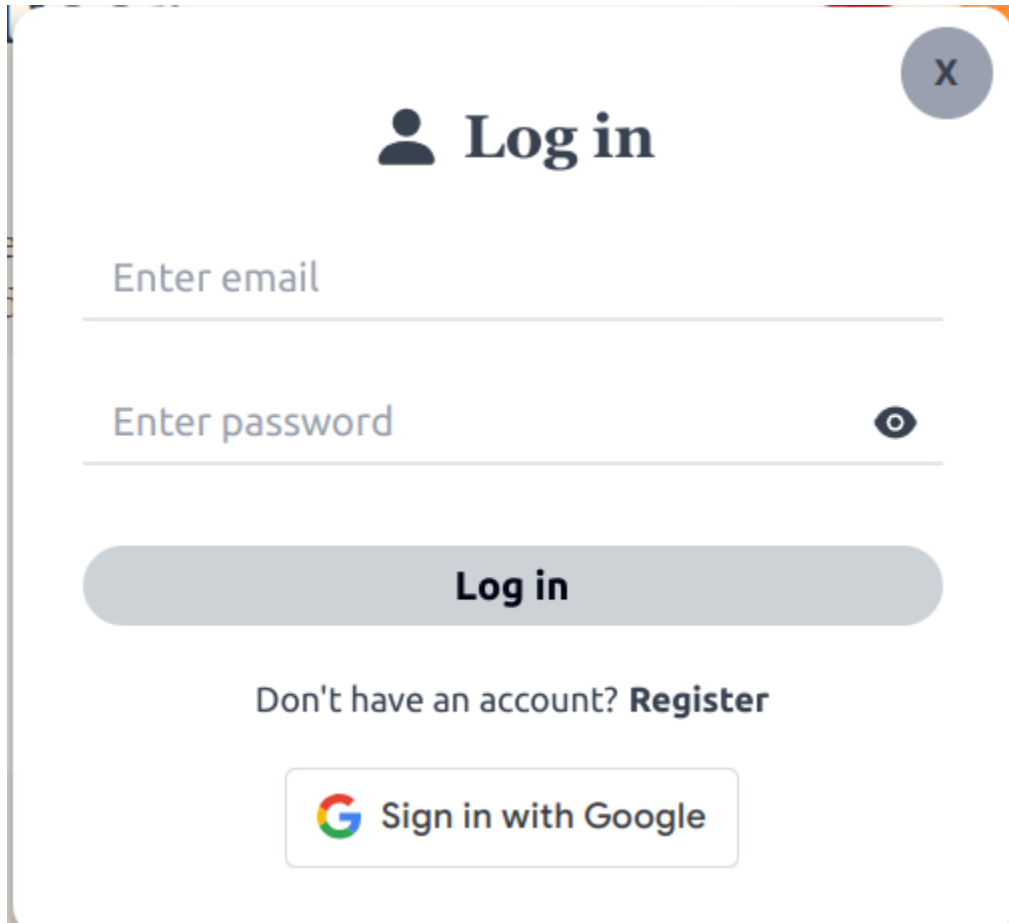




Fig : Home page

Login Page:




A login page UI mockup. At the top right is a close button (X). Below it is a header with a person icon and the text "Log in". There are two input fields: "Enter email" and "Enter password". The password field has a toggle icon (an eye) to its right. Below the inputs is a large grey button labeled "Log in". Underneath the button is the text "Don't have an account? Register". At the bottom is a button with the Google logo and the text "Sign in with Google".



 **Log in**

Enter email

Enter password 

Log in

Don't have an account? **Register**


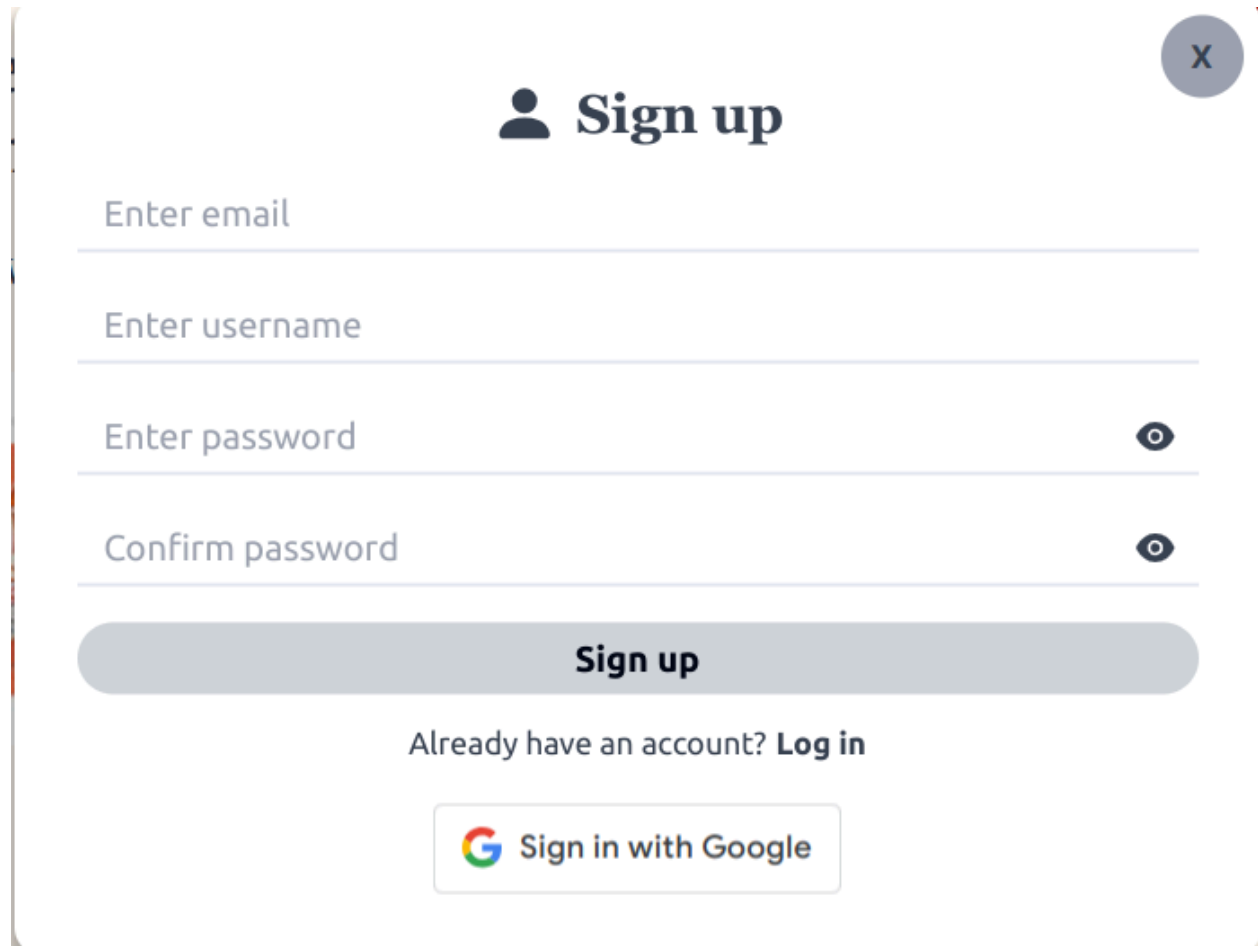

 Sign in with Google


Fig : Login page

Sign up Page:




A UI mockup of a sign-up page. At the top right is a dark grey circle with a white 'X' icon. Below it, centered, is a dark grey person icon followed by the text 'Sign up' in a bold, dark blue font. There are four input fields stacked vertically, each with a light grey border and a light grey placeholder text: 'Enter email', 'Enter username', 'Enter password', and 'Confirm password'. To the right of the 'Enter password' and 'Confirm password' fields is a dark grey eye icon. Below the input fields is a wide, rounded grey button with the text 'Sign up' in bold black font. Below the button is the text 'Already have an account? Log in' in a dark grey font. At the bottom is a white button with a rounded border, containing the Google 'G' logo and the text 'Sign in with Google' in dark grey font.


 X

 **Sign up**

Enter email

Enter username

Enter password 

Confirm password 

Sign up

Already have an account? **Log in**


 Sign in with Google

Fig : Sign up page

My Account Page:

 **Quick Food**

Profile 60% complete

 **Personal info**

 **Update Password**

 **Profile**



Id:

Username:

Address:

Mobile:

 **Edit**

Fig : My Account page

3.4.5.2 Software Interface

Browser: Chrome, Safari, Firefox, Edge

Operating System: Windows 7 or higher, Any version of MacOS, Any Version of Linux based OS.

3.4.5.3 Hardware Interface

Laptop or Mobile Device: Users access the web app using either a laptop or a mobile device such as a smartphone or tablet. These devices serve as the primary platform for interacting with the application's interface.

Input Device or Touchscreen: Users navigate the web app using input devices such as keyboards and mice for laptops or touchscreens for mobile devices. These input methods facilitate interactions such as browsing menus, placing orders, and managing accounts.

Internet Connection: Both laptops and mobile devices require an internet connection to access the food delivery web app. Users rely on Wi-Fi or cellular data connections to establish communication with the application servers hosted on remote cloud infrastructure. Stable and high-speed internet connections ensure smooth browsing, fast order processing, and real-time updates.

4 Supporting Information

This document does not need any supporting information from other documents. All the diagrams presented in this document were created with LucidChart and Creately Online Diagram maker.