ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who contributed to the development and success of the Foodie system.

First and foremost, we extend our appreciation to our project manager for their leadership, guidance, and unwavering support throughout the project lifecycle. Their dedication and commitment played a pivotal role in steering the project towards success.

We are also thankful to our talented team of developers, designers, and testers for their hard work, creativity, and expertise in bringing the Foodie system to life. Each team member's contribution, whether in frontend development, backend logic, UI/UX design, database management, quality assurance, or system administration, has been invaluable to the project's accomplishment.

Furthermore, we would like to acknowledge the invaluable feedback and support received from our stakeholders and end-users. Your insights, suggestions, and feedback have been instrumental in shaping the Foodie system and ensuring it meets the needs and expectations of its users.

Last but not least, we express our gratitude to our families, friends, and loved ones for their understanding, patience, and encouragement throughout the project journey.

Thank you all for your dedication, collaboration, and commitment to excellence. Together, we have created a system that we can all be proud of.

Table of Contents

01. Introduction

The Foodie project is a comprehensive food ordering system developed using Java and SQL, with the NetBeans IDE serving as the primary development platform. Leveraging the Swing package for its graphical user interface, the system is intuitively designed using Java JFrame classes, ensuring seamless navigation and interaction for users. The project's primary objective is to streamline the process of food ordering, facilitating convenient and efficient transactions for both customers and vendors. In the context of object-oriented programming, Foodie stands as a quintessential example of applying OOP principles to real-world scenarios, emphasizing modularity, encapsulation, and extensibility to enhance code maintainability and scalability. Its importance lies in demonstrating how object-oriented design can be utilized to create robust, user-friendly applications tailored to specific domains like food service.

02. Project Description

- Detailed explanation of the project's functionality.

The Foodie project is a comprehensive food ordering system designed to streamline the process of ordering food for both customers and vendors. Here's a detailed explanation of its functionality:

1. User Sign in and Sign up: Foodie allows users to create accounts by providing necessary details like name, email, and password. Once registered, users can log in securely using their credentials.
2. Order Now page: Users can browse through the menu items. They can add them to their cart and proceed to checkout. Foodie supports only for payment method, cash on delivery.
3. Admin Dashboard: The admin dashboard provides administrators with insights into the overall performance of the system. Admins can view sales reports, user feedbacks & user details.
4. Feedback and About us Page: After receiving their orders, users can provide feedback and ratings based on their experience. This helps maintain the quality of service and allows other users to make informed decisions when placing orders. This page also include restaurant's mobile number, address & email.

Overall, Foodie aims to provide a seamless and convenient food ordering experience for both customers and vendors, leveraging the power of Java, SQL, and Swing to deliver a robust and user-friendly solution.

- Description of the problem domain and how the project addresses it.

The Foodie project addresses inefficiencies and complexities in traditional food ordering processes for customers and vendors. It centralizes the ordering process by providing a single platform for seamless menu browsing. It also provides limited menu visibility, mitigates order inaccuracy and miscommunication, and automates order confirmation and tracking. Foodie also offers a digital platform for vendors to manage their menus and track orders, ensuring timely fulfillment. It also facilitates customer engagement through feedback and rating features, allowing users to share experiences and provide valuable insights to vendors. This project streamlines the process, improving efficiency, accuracy, and overall satisfaction.

- Mention of any specific features or functionalities implemented.

One specific feature implemented in the Foodie project is the ability for customers to view their past orders categorized by date using a JComboBox in Java. This feature enhances user experience by providing a convenient way for customers to track their order history and easily access information about previous transactions.

Here's how this functionality works:

1. Order History Retrieval: When a customer logs into their account, Foodie retrieves their order history from the database, including details such as order ID, date of order, items ordered, and order status.
2. Categorization by Date: Foodie organizes the order history into categories based on the date of each order. This categorization allows customers to quickly locate orders from specific time periods, making it easier to manage and review their past transactions.
3. Display Using JComboBox: The order history is presented to the customer using a JComboBox component in the graphical user interface. Customers can select a specific date from the dropdown menu to filter and display orders placed on that date.
4. Detailed Order Information: Upon selecting a date from the JComboBox, Foodie displays detailed information about the orders placed on that particular date, including the list of items ordered, total cost, and order status.

By implementing this feature, Foodie enhances user engagement and satisfaction by providing customers with convenient access to their order history, facilitating better organization and management of past transactions.

03. Technologies Used

- List of programming languages, frameworks, and tools used in the project.

1. Programming Languages:
    - Java: The primary programming language used for backend logic, frontend GUI development, and overall application logic.
    - SQL: Used for database management and querying to store and retrieve data related to users, orders, menus, etc.
2. Frameworks:
    - Swing: A Java GUI toolkit used for developing the graphical user interface (GUI) of the application. It provides components such as buttons, text fields, and panels for building interactive UIs.
    - JDBC (Java Database Connectivity): A Java API used to interact with relational databases like MySQL or PostgreSQL. It enables the Foodie application to execute SQL queries and manage database connections.
3. Integrated Development Environment (IDE):
    - NetBeans IDE: Chosen as the development environment for its robust support for Java development, including features like code editing, debugging, and project management. NetBeans simplifies the development process by providing tools specifically tailored for Java projects.

4. Database Management System (DBMS):
    - MySQL: A popular open-source relational database management system used to store and manage data related to users, orders, menus, etc. MySQL offers scalability, reliability, and performance, making it suitable for web applications like Foodie.
5. Version Control:
    - Git: A distributed version control system used to track changes to the project codebase, collaborate with team members, and manage different versions of the software. Git facilitates code management and ensures the integrity of the project's source code.
6. UI Design:
    - Foodie project also utilizes Figma for design and prototyping purposes. Figma is a cloud-based design tool that enables collaboration among team members in creating user interface (UI) designs, wireframes, and interactive prototypes.

- Version numbers if applicable.

1. Java – 21.0.2
2. Sql – mysql 5x
3. Git – v3

04. Timeline of Work

- Chronological breakdown of the project development process.
    1. Project Initiation:
        - Define project scope, objectives, and requirements.
        - Conduct market research and competitor analysis.
        - Establish project timeline and milestones.
    2. Requirement Gathering and Analysis:
        - Gather user requirements through stakeholder meetings, surveys, and interviews.
        - Analyze requirements to identify key features, functionalities, and system constraints.
        - Create user stories, use cases, and requirement specifications.
    3. Design Phase:
        - Design system architecture, including frontend and backend components.
        - Create wireframes, mockups, and UI/UX designs using tools like Figma.
        - Define database schema and data models.
        - Plan integration with external services (e.g., payment gateway).
    4. Implementation:
        - Set up development environment and version control system (e.g., Git).
        - Develop frontend components using Java Swing for the user interface.
        - Implement backend logic using Java programming language, incorporating object-oriented principles.
        - Integrate with SQL database (e.g., MySQL) for data storage and retrieval.
        - Develop admin dashboard functionalities for managing orders, users, and feedback.

- Start and end dates of major milestones.
    1. Project Initiation:
        - Start Date: February 19, 2024
        - End Date: February 24, 2024
    2. Requirement Gathering and Analysis:
        - Start Date: February 25, 2024
        - End Date: March 5, 2024
    3. Design Phase (UI):
        - Start Date: March 6, 2024
        - End Date: March 12, 2024
    4. Implementation:
        - Start Date: March 12, 2024
        - End Date: March 19, 2024
    5. Testing & Fixing bugs:
        - Start Date: March 19, 2024
        - End Date: April 1, 2024

- Allocation of tasks among team members if applicable.

    1. Hirunima – Sign In and Sign Up Page
    2. Rasun – Order Now Page and UserDetails page of Admin Panel
    3. Supun – My Orders Page and UserFeedback page of Admin Panel
    4. Oshini – Feedback page and Dashboard page of Admin Panel
    5. Navoda – Home Page and CustomerOrders page of Admin Panel

05. System Architecture

- Overview of the system's architecture.

The frontend of a restaurant website is a user interface developed using Java Swing, allowing users to sign in, browse menu items, add them to the cart, and provide feedback. It also handles user input handling, communication with the server, and fetching menu items. The server-side backend manages user authentication, order processing, feedback handling, and admin dashboard functionalities. It uses a relational database like MySQL for storage and JDBC for query execution. API endpoints are used to handle client requests and enforce business rules for data integrity and security. The integration layer facilitates data exchange between the frontend and backend components, while error handling ensures robustness and reliability. The database layer defines tables for storing user accounts, menu items, orders, feedback, and admin-related data, and manages data persistence to ensure secure storage and retrieval.

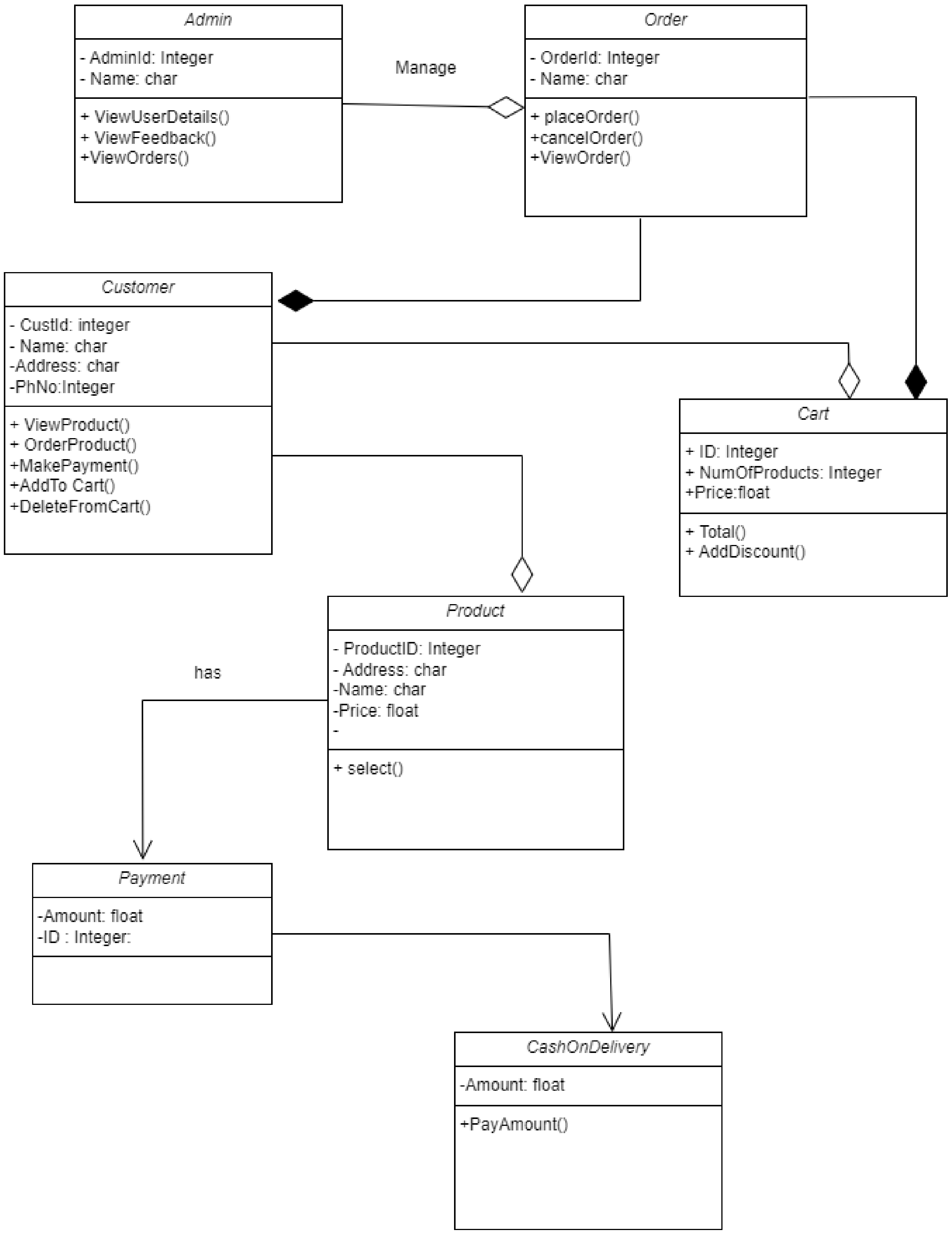
- High-level description of classes, modules, and their relationships.\

1. HomePage.java:

    - Description: Represents the homepage of the Foodie system where users initially land.
    - Relationships: Acts as the entry point of the application. May contain UI components for navigating to other pages like login, order now, or about us/feedback.
2. Login.java:
    - Description: Provides a page for users to log into the system.
    - Relationships: Dependent on the HomePage class as users typically navigate to the login page from the homepage. Communicates with authentication services to verify user credentials.
3. MyOrders.java:
    - Description: Displays the orders categorized according to the date for a specific user.
    - Relationships: May depend on authentication services to ensure only authenticated users can view their orders. Interacts with order management services to fetch and display order history.
4. NewJFrame.java:
    - Description: Combines the About Us and Feedback page functionalities.
    - Relationships: May have UI components for displaying information about the restaurant (About Us) and capturing feedback from users. Could interact with feedback management services to submit user feedback.
5. OrderNowPage.java:
    - Description: Represents the page where users can browse menu items, add them to the cart, and proceed to checkout.
    - Relationships: Likely dependent on menu management services to fetch and display menu items. May interact with order management services to handle order placement.
6. SignInAndSignUp.java:
    - Description: Provides a page for users to register/sign up for the system.
    - Relationships: May have dependencies on authentication services for user registration. Typically navigated to from the login page for new users or users without an account.
7. Dashboard.java:
    - Description: Navigation bar to other pages of the admin panel.
    - Relationships: Likely acts as a container for navigation links/buttons to different admin-related pages like FeedbackAdmin and OrdersAdmin.
8. FeedbackAdmin.java:
    - Description: Displays user feedbacks.
    - Relationships: May interact with feedback management services to fetch and display user feedback.
9. OrdersAdmin.java:
    - Description: Displays total sales and items sold, categorized according to customer name.
    - Relationships: May interact with order management services to fetch and display total sales and items sold. Likely contains UI components such as JComboBox for categorizing orders by customer name.
10. UserDetails.java:

- Description: Stores user details such as first name, last name, username, password, and email.
- Relationships: Likely used for user registration and authentication. May interact with authentication services for user management.

- UML diagrams depicting class relationships, if created.

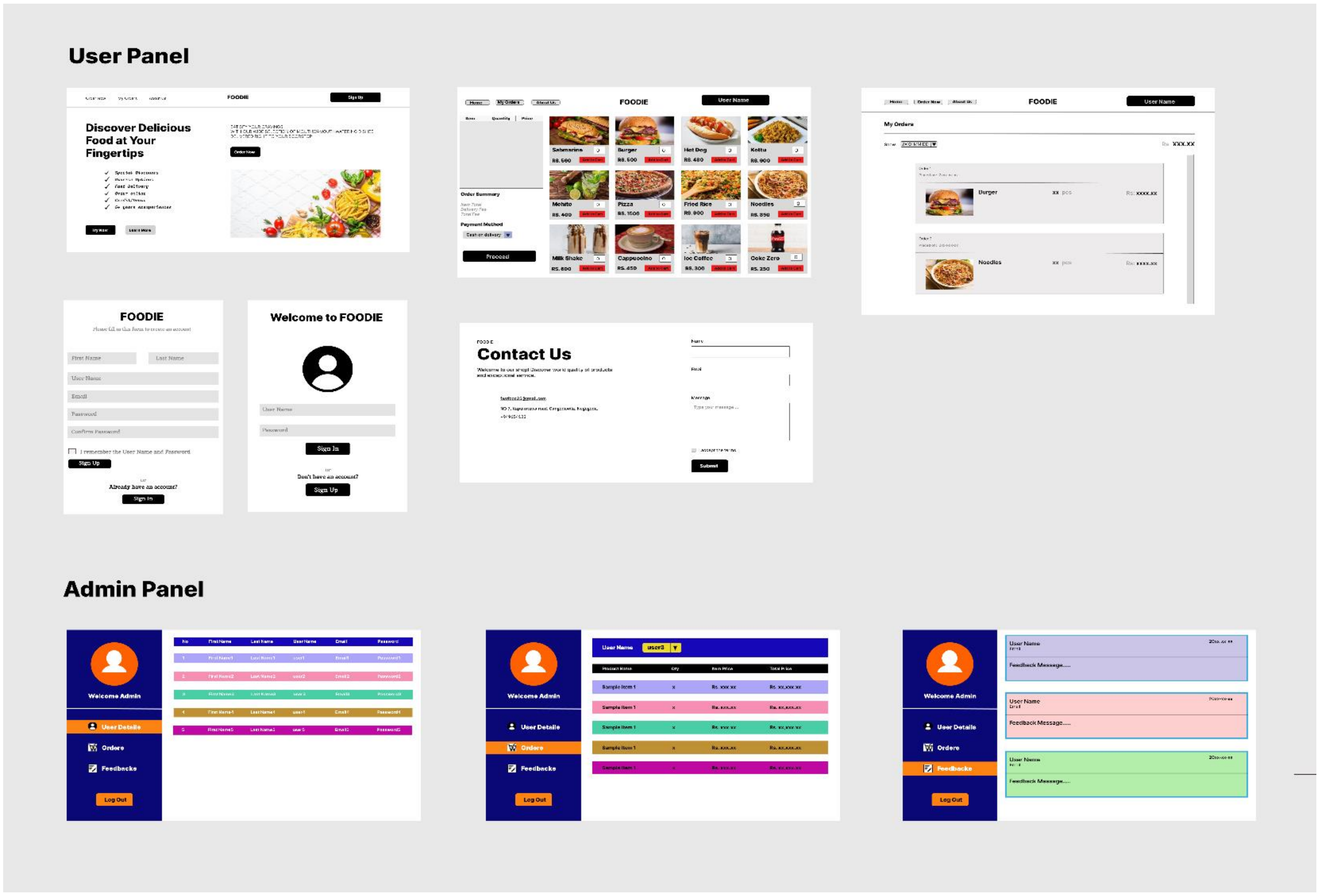


06. Interface Design

- Description of the user interface components.

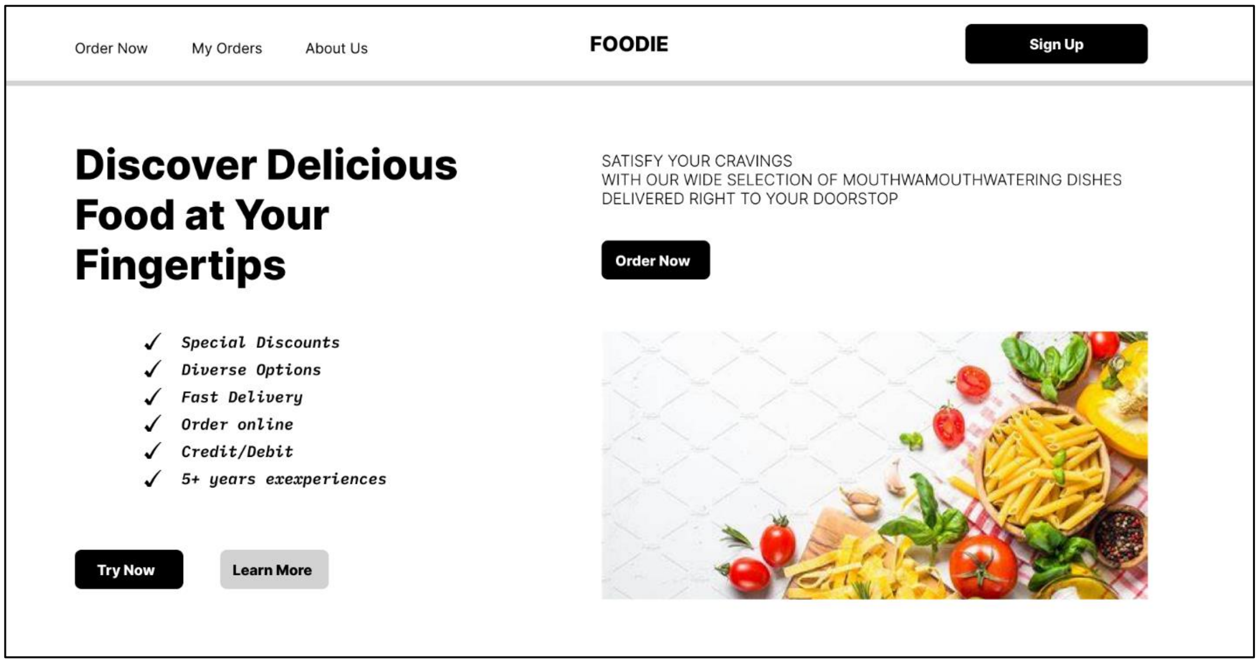
The Foodie system's user interface components are designed to offer a seamless and intuitive experience for users navigating through various functionalities. The home page features a navigation bar providing access to login, order now, about us/feedback, and admin dashboard sections. Upon logging in, users can access their order history, categorized by date, and browse menu items on the order now page, where they can add items to their cart and proceed to checkout. The sign-in/sign-up page allows new users to register, while existing users can log in securely. Admins have access to a dashboard displaying sales reports, user feedback, and order summaries. Additionally, users can provide feedback on their experience via a feedback form, while the about us section provides restaurant contact details. These components collectively enhance user engagement and streamline interactions within the Foodie system.

- Mockups or wireframes of the user interface, if available.



- Screenshots of the interface showcasing different functionalities.

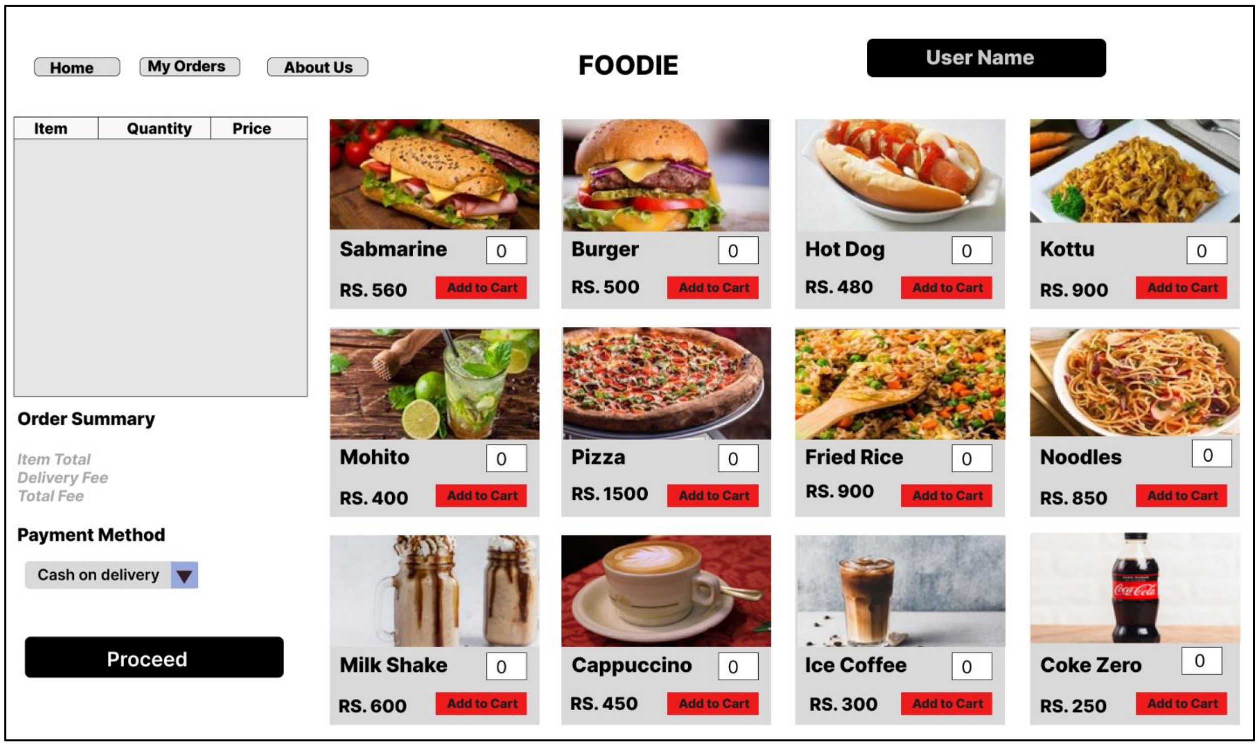
  Figma File Link - https://www.figma.com/file/suucLQRSNXi5dvJduDqNE9/Untitled?type=design&node-id=0%3A1&mode=design&t=SlR82ifnc3igoViU-1



Discover Delicious Food at Your Fingertips

- ✓ *Special Discounts*
- ✓ *Diverse Options*
- ✓ *Fast Delivery*
- ✓ *Order online*
- ✓ *Credit/Debit*
- ✓ *5+ years exexperiences*

SATISFY YOUR CRAVINGS
WITH OUR WIDE SELECTION OF MOUTHWAMOUTHWATERING DISHES
DELIVERED RIGHT TO YOUR DOORSTOP



Item	Quantity	Price

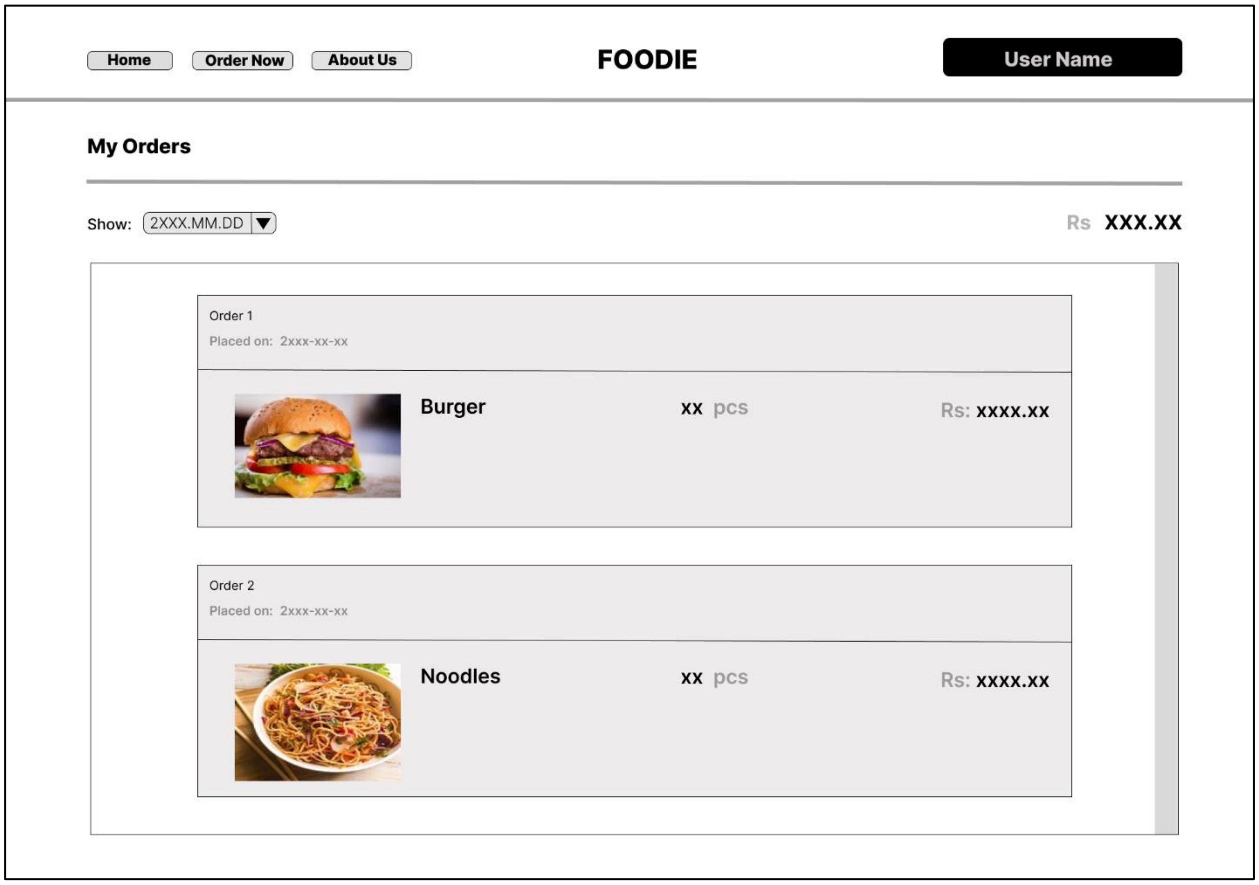
Order Summary

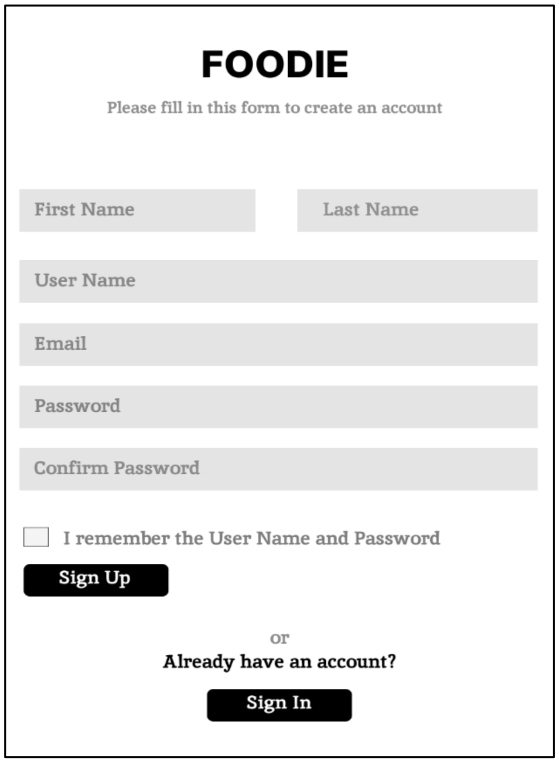
Item Total
Delivery Fee
Total Fee

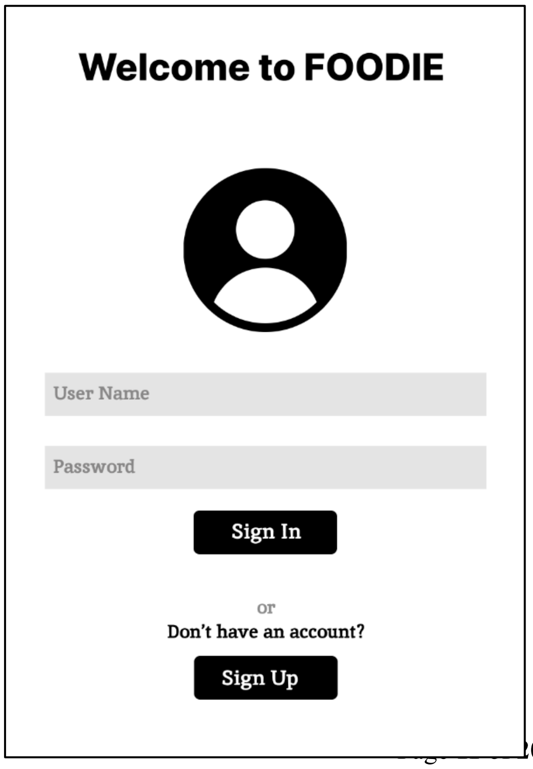
Payment Method

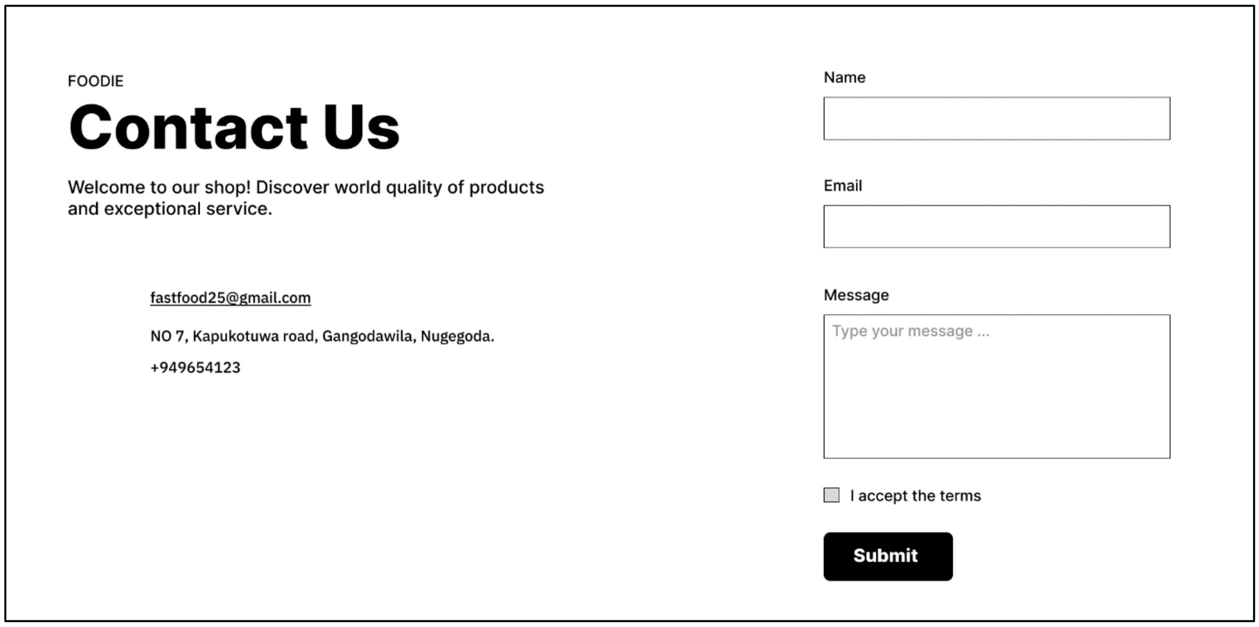
Cash on delivery

Sabmarine 0 RS. 560 Add to Cart	Burger 0 RS. 500 Add to Cart	Hot Dog 0 RS. 480 Add to Cart	Kottu 0 RS. 900 Add to Cart
Mohito 0 RS. 400 Add to Cart	Pizza 0 RS. 1500 Add to Cart	Fried Rice 0 RS. 900 Add to Cart	Noodles 0 RS. 850 Add to Cart
Milk Shake 0 RS. 600 Add to Cart	Cappuccino 0 RS. 450 Add to Cart	Ice Coffee 0 RS. 300 Add to Cart	Coke Zero 0 RS. 250 Add to Cart


Home
Order Now
About Us
FOODIE
User Name
My Orders
Show: 2XXX.MM.DD
Rs XXX.XX
Order 1
Placed on: 2xxx-xx-xx
Burger
xx pcs
Rs: xxxx.xx
Order 2
Placed on: 2xxx-xx-xx
Noodles
xx pcs
Rs: xxxx.xx


FOODIE
Please fill in this form to create an account
First Name
Last Name
User Name
Email
Password
Confirm Password
I remember the User Name and Password
Sign Up
or
Already have an account?
Sign In


Welcome to FOODIE
User Name
Password
Sign In
or
Don't have an account?
Sign Up



FOODIE

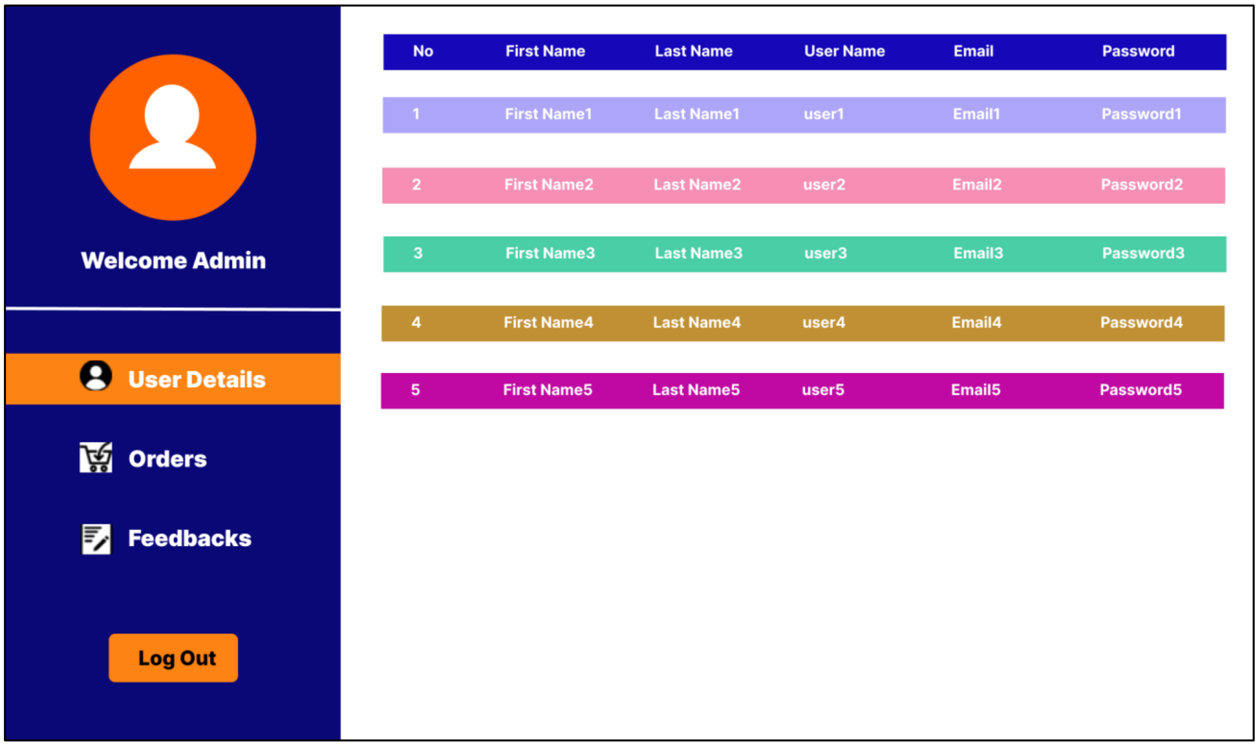
Contact Us

Welcome to our shop! Discover world quality of products and exceptional service.

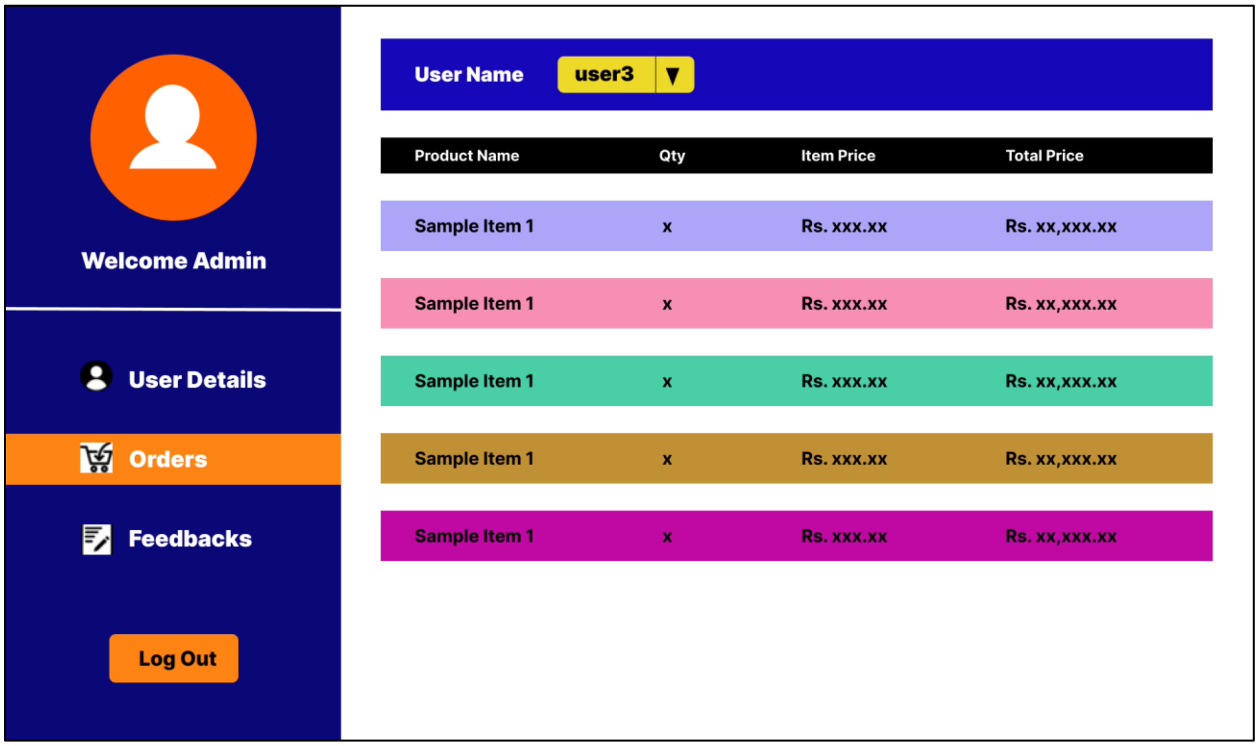
fastfood25@gmail.com

NO 7, Kapukotuwa road, Gangodawila, Nugegoda.

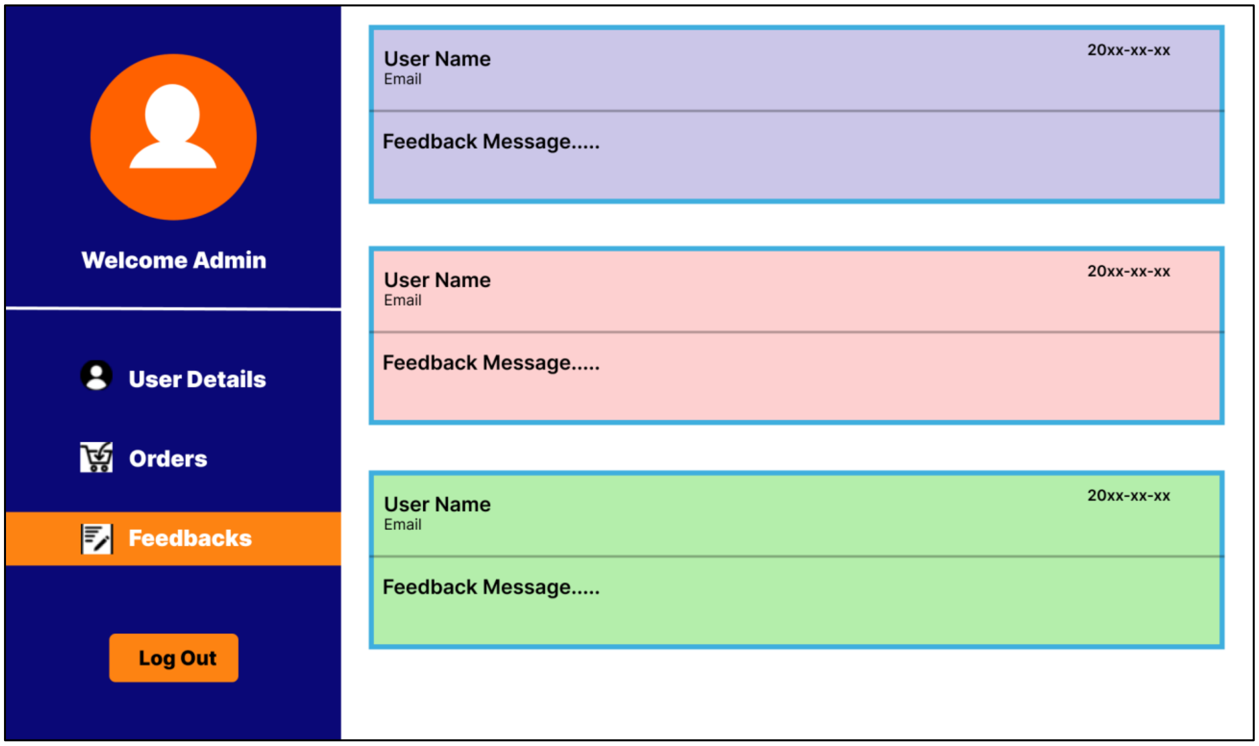
+949654123



No	First Name	Last Name	User Name	Email	Password
1	First Name1	Last Name1	user1	Email1	Password1
2	First Name2	Last Name2	user2	Email2	Password2
3	First Name3	Last Name3	user3	Email3	Password3
4	First Name4	Last Name4	user4	Email4	Password4
5	First Name5	Last Name5	user5	Email5	Password5



Product Name	Qty	Item Price	Total Price
Sample Item 1	x	Rs. xxx.xx	Rs. xx,xxx.xx
Sample Item 1	x	Rs. xxx.xx	Rs. xx,xxx.xx
Sample Item 1	x	Rs. xxx.xx	Rs. xx,xxx.xx
Sample Item 1	x	Rs. xxx.xx	Rs. xx,xxx.xx
Sample Item 1	x	Rs. xxx.xx	Rs. xx,xxx.xx



User Name 20xx-xx-xx
Email
Feedback Message.....

User Name 20xx-xx-xx
Email
Feedback Message.....

User Name 20xx-xx-xx
Email
Feedback Message.....

07. Implementation Details

- **Key Algorithms and Data Structures Used**
    - ✓ **Menu Management**

In the Foodie project, efficient menu management is crucial for providing a seamless food ordering experience. algorithms and data structures play a role in this project.

ArrayList -: Menu items could be stored in an ArrayList data structure. ArrayLists offer dynamic resizing, allowing for the flexible addition and removal of menu items. This data structure enables quick retrieval of menu items based on index, facilitating efficient browsing for users.

✓ **User Authentication**

Ensuring secure user authentication is paramount in any application handling sensitive user data. Algorithms and data structures are used for user authentication in Foodie.

Database Tables -: User credentials, including usernames and passwords, are typically stored in a database table. This structured data storage allows for efficient retrieval and validation of user credentials during the authentication process.

Arrays -: Arrays employed to store user data during authentication. For instance, you can have an array where each element represents a user account, containing information such as username, password, email, etc. This array can be searched efficiently to validate user credentials during login.

- **Design Patterns or Principles Applied**

Object-Oriented Programming (OOP) principles are fundamental guidelines that help in designing software systems that are modular, maintainable, and scalable. Some key OOP principles and how they can be applied in our Foodie project.

✓ **Encapsulation**

Encapsulation is the bundling of data and methods that operate on the data into a single unit (class). This principle helps in hiding the internal state of objects and restricting direct access to it. In Foodie, Encapsulate attributes like name and price within a Menu Item class. Use private access modifiers for these attributes and provide public getter and setter methods to access and modify them.

✓ **Abstraction**

Abstraction involves simplifying complex systems by modeling only the essential aspects and hiding unnecessary details. In Foodie, Define a Menu interface to abstract common functionalities like adding items, removing items, and displaying items. Implement this interface in different classes representing specific menu categories.

- **Challenges Faced During Implementation and Solutions**

✓ **Limited Payment Method**

Challenge -: We only offer Cash on Delivery (COD) as a payment option because we don't have an online payment gateway.

Solution -: Since we don't have an online payment system, we simplify things by only offering Cash on Delivery. This way, users don't have to worry about online transactions, making it easier and more convenient for them to order food.

✓ **Time Constraints**

Challenge -: There's limited time available for developing the Foodie project. Limited time may lead to rushed development, potential errors, or incomplete features.

Solution -: Prioritize essential features and allocate time wisely. By prioritizing tasks and managing time effectively, the Foodie project can ensure that essential features are completed despite time constraints.

08. Testing

- Overview of the testing approach used.

The testing approach for the Foodie system involves a comprehensive strategy encompassing various types of testing to ensure the reliability, functionality, and performance of the application. Here's an overview of the testing approach used:

1. Unit Testing:
   - Description: Individual components, such as classes and methods, are tested in isolation to verify their correctness and behavior.
   - Implementation: Unit tests are written using testing frameworks like JUnit for Java to validate the functionality of critical components such as authentication services, order processing, and menu management.
2. Integration Testing:
   - Description: Validates the interactions and integration between different modules and components of the system.
   - Implementation: Integration tests are conducted to verify that modules such as user authentication, order management, and feedback submission work together seamlessly, ensuring data flow and communication between components is smooth.
3. Functional Testing:
   - Description: Verifies that the application functions correctly according to its specifications and requirements.
   - Implementation: Functional tests are performed to validate user interactions, such as logging in, browsing menus, placing orders, and providing feedback. This ensures that all user-facing features work as expected.

- Description of test cases and scenarios.

1. User Authentication:
   - Test Case 1: Valid Login Credentials

Scenario: Enter correct username/email and password combination.

Expected Outcome: User should successfully log in and be redirected to the user dashboard.

   - Test Case 2: Invalid Login Credentials

Scenario: Enter incorrect username/email or password combination.

Expected Outcome: User should not be able to log in and should receive an error message indicating invalid credentials.

2. Menu Browsing and Ordering:

   - Test Case 3: Browse Menu Items

Scenario: Navigate to the menu page and verify the display of various menu categories and items.

Expected Outcome: Menu items should be displayed correctly, categorized, and with accurate details.

- Test Case 4: Add Items to Cart

  Scenario: Select items from different categories and add them to the cart.

  Expected Outcome: Selected items should be added to the cart with correct quantities and prices.

- Test Case 5: Place Order

  Scenario: Proceed to checkout from the cart, enter delivery details, and complete the order.

  Expected Outcome: Order should be processed successfully, and the user should receive an order confirmation with an order ID.

3. Feedback Submission:

- Test Case 6: Provide Feedback on Order

  Scenario: After completing an order, navigate to the feedback page, select the order, and provide a rating and comments.

  Expected Outcome: Feedback should be submitted successfully for the selected order.

- Test Case 7: Feedback Without Order Selection

  Scenario: Attempt to submit feedback without selecting an order.

  Expected Outcome: Submission should be blocked, and the user should receive an error message indicating the need to select an order.

4. Admin Dashboard:

- Test Case 8: Access Admin Dashboard

  Scenario: Log in with admin credentials and navigate to the admin dashboard.

  Expected Outcome: Admin should be able to access the dashboard displaying sales reports, user feedback, and order summaries.

- Test Case 9: View User Details

  Scenario: Access the user details section from the admin dashboard and view user information.

  Expected Outcome: User details such as name, email, and order history should be displayed correctly.

- Results of testing, including any bugs or issues encountered and their resolutions.

1. User Authentication Testing:
   - Result: No significant issues were found during testing. Users were able to log in successfully with valid credentials.
   - Resolution: N/A
2. Menu Browsing and Ordering Testing:
   - Result: Some users reported occasional errors when adding items to the cart.
   - Issue: An intermittent bug was identified where certain menu items were not being added to the cart correctly.
   - Resolution: The issue was traced to a caching mechanism causing stale data to be used during cart updates. The caching mechanism was revised to ensure real-time updates, resolving the issue.
3. Feedback Submission Testing:
   - Result: Users encountered errors when submitting feedback without selecting an order.
   - Issue: Validation was missing to ensure users select an order before submitting feedback.
   - Resolution: Validation checks were implemented to prompt users to select an order before proceeding with feedback submission, preventing errors.
4. Admin Dashboard Testing:
   - Result: Admins experienced delays in loading user details and feedback data on the dashboard.
   - Issue: Inefficient database queries and data retrieval processes were causing delays in loading dashboard data.
   - Resolution: Database queries were optimized, and caching mechanisms were introduced to improve data retrieval performance, resolving the delays.

09. Future Enhancements

Ideas for future improvements or features.

- **Multiple Payment Methods** -: Introduce support for various payment methods such as credit/debit cards, digital wallets, or online payment gateways to offer users more flexibility and convenience in payment options.
- **Advanced User Profiles -:** Enhance user profiles with features like order history visualization, favorite orders, dietary preferences, and personalized recommendations based on past orders to improve user engagement and satisfaction.
- **Real-Time Order Tracking -:** Implement real-time order tracking functionality, allowing users to track the status of their orders from preparation to delivery. Integration with delivery services' APIs can provide live updates on order location and estimated delivery time.
- **Integration with Third-Party APIs -:** Integrate with popular food delivery platforms or restaurant reservation services to expand the range of options available to users and provide a comprehensive food ordering experience.

<u>Areas identified for optimization or refinement.</u>

- **Performance Optimization** -: Identify and optimize performance bottlenecks, such as database queries or resource-intensive operations, to improve overall system responsiveness and scalability.
- **User Interface Refinement** -: Continuously refine the user interface based on user feedback and usability testing. Ensure consistency in design elements, layout, and navigation to provide a seamless and intuitive user experience across different devices and screen sizes.
- **Security Enhancements** -: Strengthen security measures by implementing additional layers of protection against common vulnerabilities such as SQL injection, cross-site scripting (XSS), and session hijacking. Regular security audits and updates are essential to safeguard user data and prevent unauthorized access

10. Link to Project

- Upload all project files to the drive. Share the link to the folder in this section (**with view and download access!)**
  Link -
  https://drive.google.com/drive/folders/1RZ2jSzIS9JZnTV5M5q4vG01NGKoZyAx9?usp=sharing

- Share the Git repository (**with suitable view access!)**
  Link - https://github.com/HarleeS/Food-ordering-project.git
  (The project is in branch called main not the master branch)

11. Conclusion

- **<u>Summary of Key Achievements and Learning:</u>**

- Our effort to construct a food ordering system with Java and the NetBeans Swing package has been an interesting achievement, showing our capacity to design useful software. We were able to obtain practical software development experience through this project, specifically in the areas of event handling and GUI design. We were able to create an interactive and customer friendly user interface by implementing Swing components, which made it possible for users to easily navigate menus, select food items and place orders. Additionally, incorporating backend features like payment processing and order management improved our knowledge of system integration and software architecture. All things considered, this project has greatly advanced our understanding of Java programming and GUI development, giving us useful skills for upcoming software projects.

- **<u>Reflection on the Overall Experience and Outcomes:</u>**

- The process of creating a food ordering system has been rewarding and difficult, giving us priceless knowledge and experiences. Through teamwork, we discovered how important efficient coordination and communication are to project management. Overcoming obstacles like debugging mistakes and improving functionality expanded our ability to solve problems and keep going in the when we faces difficulties. This project also provided us with a platform for ongoing learning and development, inspiring us to investigate new software development techniques and technologies. As we move forward, we are sure that we will be able to use the information and abilities we gained from this project to pursue excellence in software engineering and innovation in the future.

12. References

We used the Java swing package in the Netbeans software and in order to complete the project we used external Java libraries such as

1. **JDatePicker**: With the help of this library, we could choose dates in our application with ease. It offers a collection of date selection components. When choosing delivery dates or organizing orders, it was very helpful.
2. **JFreeChart**: A popular library for making graphs and charts in Java applications is JFreeChart. With outr application we used it to display sales data, order statistics, and other relevant information.
3. **Apache Commons IO:** This Java package offers utilities for common input/output operations. It helped with resource management, input/output operations, and file reading and writing in the application.
4. **Apache Commons Lang:** This Java programming library provides a collection of utility classes for frequent tasks. It has classes for manipulating objects, strings, and other common operations which can help to simplify the development and improve the code.

13. Appendix

- Additional materials such as code snippets, diagrams, or documentation not included in the main sections.