Food Ordering System

Dharmsinh Desai University

Academic Year: 2022-23
Department: Faculty of Management & Information Science

Subject: Software Engineering

Full Name: Alyani Mamad
Roll No: MA003
ID No : 22MAPBG029

Submitted to
Prof. Minal Shah
MCA Department

Student Sign:

Professor Sign

TABLE OF CONTENTS

Food Ordering System

- ## Introduction

★ This document serves as the official software engineering documentation for the Food Ordering System. The purpose of this document is to provide a comprehensive overview of the system, including its functional and non-functional requirements, design and architecture, implementation and testing details, deployment and maintenance plans, and future work and improvements.

★ The Food Ordering System is an online platform that enables users to order food from restaurants and cafes in their area. The system allows users to browse different restaurants and menus, select their desired items, place orders, and pay online. The system also provides features for restaurant owners and managers to manage their menus, orders, and payments, as well as for delivery drivers to track and fulfill orders.

★ The scope of the Food Ordering System includes the following key features:

- User registration and authentication
- Restaurant and menu management
- Order placement and payment processing
- Delivery tracking and fulfillment
- Feedback and review system

★ The system is designed using a client-server architecture, where the front-end is implemented using React.js and the back-end is implemented using Node.js and Express.js. The system uses a RESTful API for communication between the client and the server, and a MongoDB database for storing and retrieving data.

★ The key stakeholders of the Food Ordering System include the following:

- Users: The primary users of the system are customers who want to order food from restaurants and cafes.
- Restaurant owners and managers: The system also caters to the needs of restaurant owners and managers who want to manage theirmenus, orders, and payments.
- Delivery drivers: The system provides features for delivery drivers

who want to track and fulfill orders.

- Administrators: The system includes administrators who are responsible for managing the system, including user accounts, restaurant accounts, and order fulfillment.

★ Overall, this document provides a detailed account of the Food Ordering System, its requirements, design, implementation, and maintenance plans, as well as its future work and improvements. The document is intended for software developers, system architects, project managers, and other stakeholders who are involved in the development and maintenance of the system.

Requirements analysis and specification

Functional Requirements

The functional requirements of the Food Ordering System are as follows:

1. User Registration and Authentication:
    - Users should be able to create a new account by providing their email address, name, and password.
    - Users should be able to log in to their account using their email address and password.
    - The system should verify the user's email address before allowing them to log in.
2. Restaurant and Menu Management:
    - Restaurant owners and managers should be able to create a new restaurant account and manage their restaurant's menu.
    - Restaurant owners and managers should be able to add, delete, and modify menu items and their prices.
    - Restaurant owners and managers should be able to set their restaurant's opening and closing hours.
    - Restaurant owners and managers should be able to view their restaurant's order history.
3. Order Placement and Payment Processing:
    - Users should be able to browse different restaurants and menus and select their desired items.
    - Users should be able to place an order by selecting their delivery address, payment method, and desired delivery time.
    - The system should verify the availability of the selected items and confirm the order with the user.
    - The system should process the user's payment using a secure payment gateway.
    - The system should send an order confirmation email to the user and the restaurant.
4. Delivery Tracking and Fulfillment:
    - Delivery drivers should be able to view their assigned orders and track the delivery status in real-time.
    - Delivery drivers should be able to mark an order as delivered once they have delivered it to the user.
    - Users should be able to view the delivery status of their order in

real-time.

5. Feedback and Review System:
    - Users should be able to rate and review their orders and restaurants.
    - Users should be able to view the ratings and reviews of other users for a given restaurant.

Non-functional Requirements

The non-functional requirements of the Food Ordering System are as follows:

1. Performance:
    - The system should be able to handle a large number of users and orders simultaneously without significant performance degradation.
    - The system should respond to user requests within 3 seconds.
2. Scalability:
    - The system should be designed to scale horizontally to handle increasing traffic and user demand.
    - The system should be able to add more servers and resources as needed.
3. Security:
    - The system should use SSL/TLS encryption to secure all communication between the client and the server.
    - The system should use secure payment gateways to process payments.
    - The system should use secure authentication mechanisms to prevent unauthorized access.
4. Usability:
    - The system should have a user-friendly and intuitive interface that is easy to navigate and understand.
    - The system should provide helpful error messages and notifications to guide users through the ordering process.

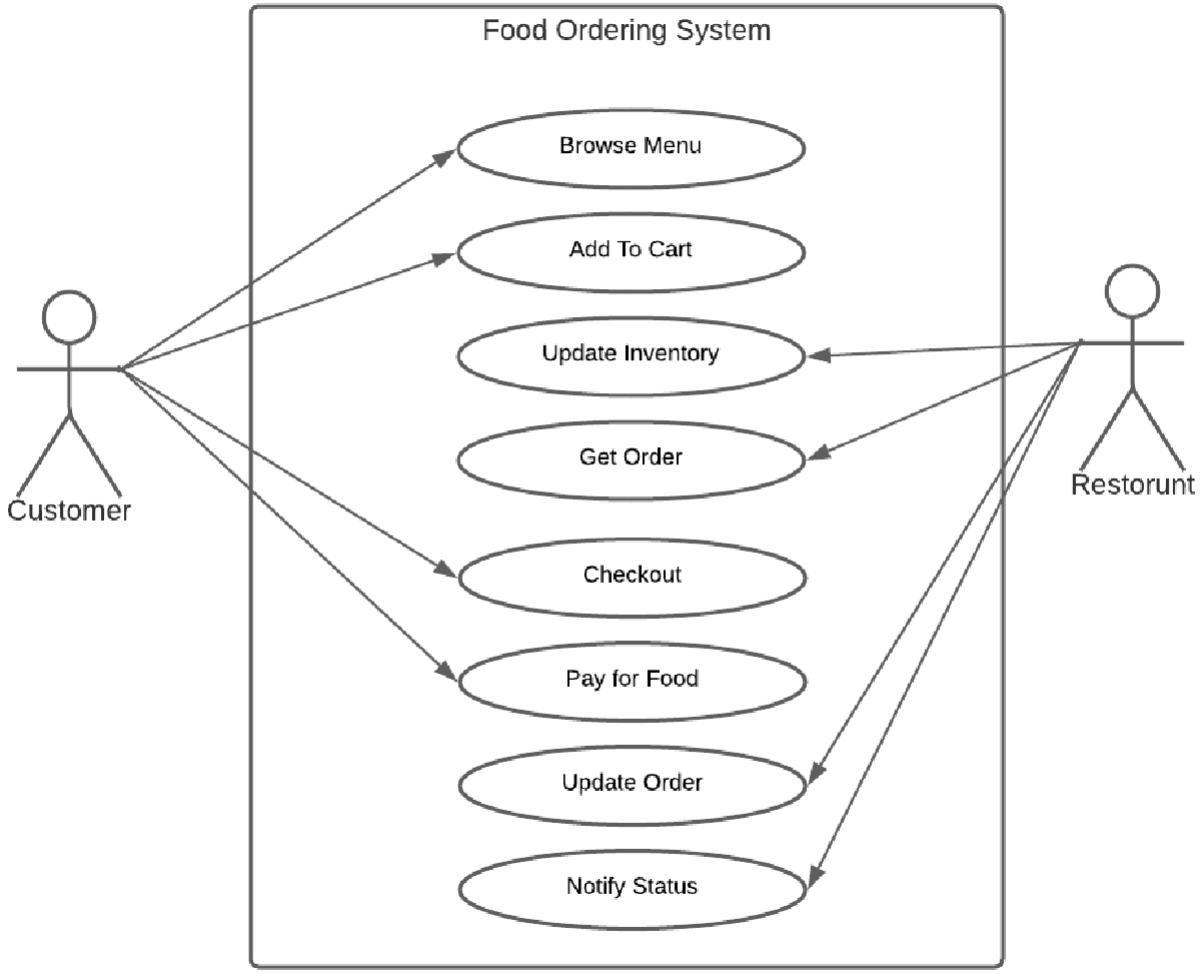
Use Cases and Scenarios

The following use cases and scenarios describe the main interactions between users and the Food Ordering System:

1. User Registration:
    - User creates a new account by providing their email address, name,

and password.
    - System verifies the user's email address and sends a confirmation email.
    - User clicks on the confirmation link and verifies their account.
2. Order Placement:
    - User browses different restaurants and menus and selects their desired items.
    - User provides their delivery address, payment method, and desired delivery time.
    - System verifies the availability of the selected items and confirms the order with the user.
    - System processes the user's payment and sends an order confirmation email to the user and the restaurant.
3. Delivery Fulfillment:
    - Delivery driver logs in to their account and views their assigned orders.
    - Delivery driver delivers the order to the user and marks it as delivered in the system.
    - User receives the order and confirms its delivery.
4. Restaurant Management:
    - Restaurant owner logs in to their account and manages their restaurant's menu.
    - Restaurant owner adds, deletes, and modifies menu items and their prices.
    - Restaurant owner views their restaurant's order history and ratings.

USE CASE DIAGRAM:



User Interface Design

The user interface of the Food Ordering System should be designed to provide a seamless and intuitive experience for users. The interface should be easy to navigate and use, with clear and concise labels and instructions. Some key design considerations include:

1. Responsive Design:
    - The interface should be optimized for different screen sizes and devices, including desktop, tablet, and mobile.
2. Intuitive Navigation:
    - The interface should have a clear and simple navigation menu that allows users to easily access different sections of the system.
3. Visual Design:
    - The interface should use a consistent visual style and color scheme that reflects the brand and identity of the system.
    - The interface should use clear and legible typography and iconography to aid in navigation and comprehension.
4. Accessibility:
    - The interface should be designed to be accessible to users with disabilities, including those who are visually impaired or have limited mobility.
    - The interface should conform to accessibility standards such as WCAG 2.1.

- ## System design

High-Level Design

The Food Ordering System will be built using a client-server architecture, with a web-based client and a cloud-based server. The client-side will be responsible for handling user interactions and presenting information to the user, while the server-side will be responsible for processing and storing data. The system will be built using the Model-View-Controller (MVC) architectural pattern.

The key components of the system include:

1. User Interface:
    - Provides a graphical user interface for users to interact with the system.
    - Built using web technologies such as HTML, CSS, and JavaScript.
2. Application Logic:
    - Handles user interactions and data processing.
    - Built using a server-side programming language such as PHP, Python, or Node.js.
3. Data Storage:
    - Stores user and system data, including user profiles, restaurant and menu information, and order history.
    - Utilizes a cloud-based database system such as MySQL, MongoDB, or Amazon RDS.

Detailed Design

Database Schema

The Food Ordering System will use a relational database to store data. The database schema will consist of the following tables:

1. Users: Stores user profile information, including name, email, password, and contact information.
2. Restaurants: Stores restaurant information, including name, address, and contact information.
3. Menu Items: Stores menu item information, including name, description, price, and availability.
4. Orders: Stores order information, including order details, status, and payment information.

Algorithms

The Food Ordering System will utilize various algorithms to handle tasks such as order processing, delivery routing, and recommendation engines. Some key algorithms include:

1. Order Processing Algorithm:
    - Calculates the total price of an order based on the items selected and their prices.
    - Generates an order confirmation and sends it to the user and the restaurant.
2. Delivery Routing Algorithm:
    - Determines the most efficient delivery route for a driver based on the user's location and the location of the restaurant.
    - Updates the order status to reflect the estimated delivery time.
3. Recommendation Engine:
    - Recommends menu items to users based on their order history and preferences.
    - Updates menu item availability based on popularity and inventory levels.

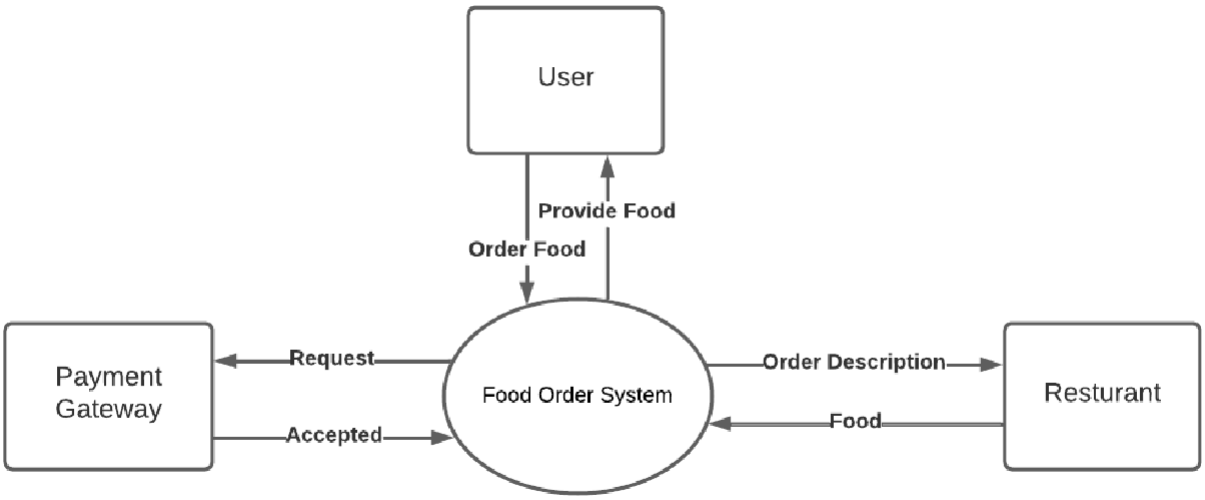
Interfaces

The Food Ordering System will utilize various interfaces to interact with external systems and APIs. Some key interfaces include:

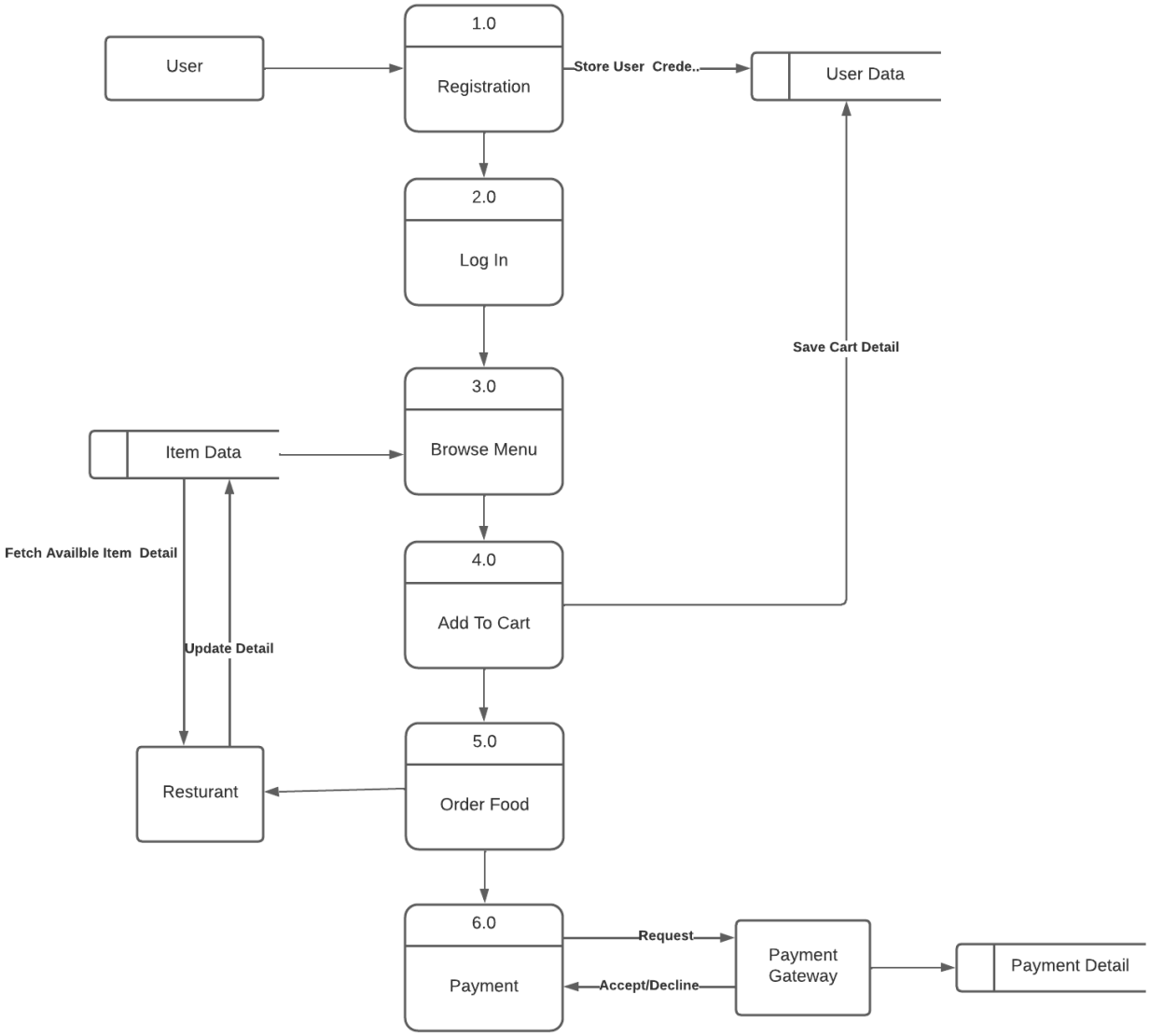
1. Payment Gateway API:
    - Integrates with a payment gateway to handle payment processing.
    - Allows users to securely enter their payment information and complete transactions.
2. Geolocation API:
    - Integrates with a geolocation API to determine the user's location and the location of the restaurant.
    - Allows for more accurate delivery routing and estimated delivery times.
3. SMS Notification API:
    - Integrates with an SMS notification API to send order confirmation and delivery updates to users and drivers.
    - Provides real-time notifications and updates to improve the user experience.

DATA FLOW DIAGRAMS:

Level 0 DFD:



Level 1 DFD:


1.0
User
Registration
Store User Crede..
User Data
2.0
Log In
Save Cart Detail
3.0
Item Data
Browse Menu
Fetch Availble Item Detail
Update Detail
4.0
Add To Cart
5.0
Resturant
Order Food
6.0
Request
Payment
Accept/Decline
Payment Gateway
Payment Detail

- ## Implementation and testing

TIMELINE CHART:

Requirements gathering and analysis	**2 Weeks**
System Design and Architecture	**3 weeks**
Database Design and Development	**2 weeks**
Front-end Development	**4 Weeks**
Back-end Development	**4 weeks**
Integration and Testing	**2 Weeks**
Deployment and Launch	**1 Weeks**

Programming Languages and Frameworks Used

The Food Ordering System will be built using modern web development technologies, including:

1. Front-end Technologies:
    - HTML, CSS, and JavaScript for building the user interface.
    - React or AngularJS for building the front-end application logic.
2. Back-end Technologies:
    - Node.js or Python for building the server-side application logic.
    - Express.js or Django for building the API endpoints.
    - MySQL or MongoDB for storing data.

Testing Strategies and Tools

The Food Ordering System will utilize various testing strategies and tools to ensure the quality and reliability of the software. Some key testing strategies and tools include:

1. Unit Testing:
    - Unit tests will be used to test individual components of the system, such as API endpoints and data processing functions.

- Jest or Mocha will be used as the testing framework.

2. Integration Testing:
   - Integration tests will be used to test the interactions between different components of the system.
   - Supertest or Postman will be used to test API endpoints.
3. User Acceptance Testing:
   - User acceptance tests will be used to test the system's functionality from a user's perspective.
   - User testing groups will be used to provide feedback and identify areas for improvement.

Agile Testing for Food Ordering System

- **Introduction :**

➢ The purpose of this Agile testing documentation is to describe the testing process and methodologies used to test the Food Ordering System. The system is designed to allow customers to order food online, make payments, and track the status of their order. The testing process will ensure that the system meets the requirements and functional specifications outlined by the stakeholders.

- **Testing Objectives :**

➢ To ensure that the system meets the functional and non-functional requirements specified by the stakeholders.
➢ To identify and report any defects or issues that affect the functionality of the system.
➢ To ensure that the system is user-friendly and easy to use.
➢ To ensure that the system is secure and can handle user data and payment information.
➢ To ensure that the system is scalable and can handle high traffic volumes.

- **Testing Scope :**

➢ Functional testing of all system features, including ordering, payment, and tracking.
➢ Non-functional testing of system performance, security, and scalability.
➢ User acceptance testing to ensure that the system meets user expectations and is easy to use.
➢ Compatibility testing to ensure that the system works on all supported devices and browsers.

- **Testing Methodologies :**

  - ➢ Agile testing methodologies, including Test-Driven Development (TDD), Behavior-Driven Development (BDD), and Continuous Integration/Continuous Deployment (CI/CD).
  - ➢ Manual testing to verify system functionality, user interface, and user experience.
  - ➢ Automated testing to verify system performance, security, and scalability.
  - ➢ Exploratory testing to identify any unforeseen issues or defects in the system.

- **Testing Enviroment :**

  - ➢ Test servers for development and testing environments.
  - ➢ Virtual machines to simulate different devices and browsers.
  - ➢ Tools for automated testing, including Selenium, JMeter, and Postman.
  - ➢ Tools for tracking and reporting defects, including JIRA, Bugzilla, or Trello.

- **Testing Process :**

  - ➢ Test planning and preparation, including defining test cases and scenarios, identifying test data, and selecting testing tools.
  - ➢ Test execution, including manual and automated testing, exploratory testing, and user acceptance testing.
  - ➢ Defect reporting and tracking, including identifying and reporting defects, assigning priorities and severity levels, and tracking defect resolution.
  - ➢ Test closure, including documenting test results, creating a test report, and providing feedback to the development team.

- **Testing Deliverables :**

  - ➢ Test plan and strategy documents.
  - ➢ Test cases and scenarios.
  - ➢ Test data and test results.
  - ➢ Defect reports and tracking logs.
  - ➢ Test summary report.

- **Conclusion :**
  - ➢ The testing process for the Food Ordering System will ensure that the system meets the requirements and functional specifications outlined by the stakeholders. By following the Agile testing methodologies, the testing process will be iterative and collaborative, allowing the development team to identify and resolve issues quickly.

Code Reviews and Testing Results

- Code reviews will be conducted regularly throughout the development process to ensure code quality and maintainability. Pull requests will be required for all code changes, and a designated reviewer will be assigned to each pull request.
- Testing results will be tracked using a test management tool, such as Jira or TestRail. Test results will be regularly reviewed and analyzed to identify areas for improvement and prioritize bug fixes. Any major issues will be addressed immediately to ensure the system remains stable and functional.

- ## Deployment and maintenance

System Deployment Plan

The Food Ordering System will be deployed using a cloud-based infrastructure, such as Amazon Web Services or Microsoft Azure. The deployment plan will consist of the following steps:

1. Provisioning: Provision the necessary infrastructure, including virtual machines, load balancers, and databases.
2. Installation: Install the necessary software and dependencies, including the web server, application server, and database server.
3. Configuration: Configure the system for the desired environment, including setting up API keys, configuring the database schema, and configuring the load balancer.
4. Testing: Conduct a series of tests to ensure the system is stable and functional, including load testing, security testing, and user acceptance testing.
5. Go-Live: Launch the system in the production environment.

System Maintenance Plan

The Food Ordering System will require ongoing maintenance to ensure it remains stable and functional. The maintenance plan will consist of the following steps:

1. Bug Fixes: Any reported bugs or issues will be addressed promptly to ensure the system remains stable and functional.
2. Updates: Regular updates will be released to improve the system's performance, security, and usability. These updates will be tested thoroughly before being released to the production environment.
3. Enhancements: The system will be continuously improved based on

user feedback and changing business requirements. These enhancements will be prioritized based on their impact and feasibility.

User Training and Support

To ensure users can effectively use the Food Ordering System, the following training and support will be provided:

1. User Documentation: Comprehensive user documentation will be provided to guide users through the system's features and functionality.
2. Onboarding Training: New users will receive onboarding training to familiarize them with the system and its features.
3. Ongoing Support: Ongoing support will be provided to users to address any questions or issues they may have while using the system.
4. Feedback Channels: Feedback channels, such as user surveys and customer support tickets, will be provided to gather user feedback and address any issues or concerns they may have.

- # Conclusion

Summary of the Project

The Food Ordering System is a web-based application that allows users to place food orders online. The system is designed to be user-friendly and easy to use, with a focus on providing a seamless ordering experience. The system includes features such as a menu, shopping cart, payment gateway, order tracking, and user account management.

Lessons Learned

During the development of the Food Ordering System, several lessons were learned, including:

1. The importance of user feedback: User feedback was critical in shaping the system's design and functionality. Regular user testing and feedback sessions helped identify areas for improvement and ensure the system met user needs.
2. The value of modular design: Modular design was essential for building a scalable and maintainable system. Breaking the system down into smaller, more manageable components allowed for easier development, testing, and maintenance.
3. The challenges of integration testing: Integration testing was one of the more challenging aspects of the development process. Coordinating the testing of multiple components required careful planning and execution.

Future Work and Improvements

The Food Ordering System is a robust and functional application, but there is always room for improvement. Some areas for future work and improvements include:

1. Mobile App: Developing a mobile application for the system would allow users to place orders more conveniently from their smartphones.
2. Personalization: Adding personalization features, such as order history and user preferences, would allow users to place orders more quickly and easily.
3. Advanced Analytics: Incorporating advanced analytics features, such as data visualization and predictive analytics, would provide valuable insights into user behavior and help identify opportunities for growth and improvement.
4. Continuous Improvements: Continuously improving the system based on user feedback and changing business requirements is essential for maintaining the system's relevance and usefulness. Regular updates and enhancements should be released to keep the system current and competitive.
5. Security: Security is of paramount importance in any web application. The Food Ordering System should be regularly audited for security vulnerabilities and any identified issues should be addressed promptly.
6. Localization: To expand the system's reach and appeal to a wider audience, localization features, such as multi-language support and currency conversion, can be added.
7. Marketing: To increase the system's visibility and attract more users, a comprehensive marketing strategy should be developed. This can include tactics such as social media advertising, email marketing, and influencer partnerships.
8. Partnership with Restaurants: Partnering with local restaurants can help expand the system's offerings and provide users with a wider selection of food options.
9. Continuous Monitoring and Improvement: The Food Ordering System should be continuously monitored and improved to ensure that it remains competitive and meets the changing needs of its users.