FOODIE

Web Application

Team Members:

- Alaa Abdalla Attia Salem
- Rewan Mohmed Saad
- Judy Wael

Project Documentation:
Food Recipe Web Application

1. Project Planning & Management

1.1 Project Proposal

The Food Recipe Web Application is a **recipe discovery and ingredient ordering platform** that allows users to explore recipes, watch cooking tutorials, and order the necessary ingredients for delivery. The system aims to simplify the cooking experience by integrating a food blog with an ingredient ordering service.

1.2 Project Plan

- **Timeline:** Development phases with milestones and deadlines.
- **Deliverables:**
    - Frontend UI (React.js)
    - Backend API (.NET Core)
    - SQL Database Implementation
    - Payment Integration
    - Deployment on AWS

1.3 Task Assignment & Roles

- **Frontend Developer:** Implement React UI, user authentication, and API integration.
- **Backend Developer:** Develop RESTful API, database management, and business logic.
- **Project Manager:** Oversee project timeline, assign tasks, and ensure quality control.

1.4 Risk Assessment & Mitigation Plan

- **Ingredient Availability Issues:** Implement real-time stock tracking.
- **Payment Gateway Failures:** Provide alternative payment methods.
- **High Traffic Handling:** Use AWS auto-scaling and caching mechanisms.

1.5 Key Performance Indicators (KPIs)

- Number of recipe views per user session.
- Order conversion rate.
- System uptime and response time.
- User engagement through ratings and reviews.

2. Literature Review

2.1 Feedback & Evaluation

- Evaluation based on functionality, usability, and performance.
- Stakeholder feedback on ease of use, design aesthetics, and system efficiency.
- Testing sessions conducted with a sample user base to assess responsiveness and user experience.

2.2 Suggested Improvements

- Enhance recipe recommendation system using AI-based user preferences.
- Improve search functionality by allowing filtering based on dietary preferences.
- Optimize payment process to include multiple secure payment gateways.
- Implement a real-time order tracking feature.
- Enhance mobile responsiveness for a seamless experience across devices.

2.3 Final Grading Criteria

- **Documentation (20%)** – Completeness, clarity, and structure of documentation.
- **Implementation (40%)** – Functionality, database efficiency, and system performance.
- **Testing (25%)** – Quality assurance, bug reports, and user acceptance testing results.
- **Presentation (15%)** – Clarity, professionalism, and demonstration of key features.

3. Requirements Gathering

3.1 Stakeholder Analysis

- **End Users:** Home cooks, food lovers, busy professionals.
- **Admin:** Manages recipes, orders, and inventory.

3.2 User Stories & Use Cases

- **User:** "As a user, I want to search for recipes based on ingredients so I can cook meals with what I have."
- **Admin:** "As an admin, I want to update ingredient availability so that users only order what is in stock."

3.3 Functional Requirements

1. User account creation and authentication.
2. Recipe browsing with filters.
3. Ingredient ordering and checkout process.
4. Video tutorial integration.
5. Admin dashboard for order and recipe management.

3.4 Non-Functional Requirements

1. Mobile-responsive design.
2. Secure payment processing.
3. Scalable hosting on AWS.
4. Fast load times and caching for optimal performance.

4. System Analysis & Design

4.1 Problem Statement & Objectives

- **Problem:** Users struggle with finding recipes and sourcing ingredients conveniently.
- **Solution:** A seamless recipe browsing and ingredient ordering system.

4.2 Use Case Diagram & Descriptions

- Includes system actors (User, Admin) and their interactions.

4.3 Software Architecture

- **Architecture Style:** Microservices with RESTful API.
- **Components:** React Frontend, .NET Backend, NoSQL Database, AWS Hosting.

5. Database Design & Data Modeling

5.1 ER Diagram (Entity-Relationship Diagram)

- Tables: Users, Recipes, Ingredients, Orders, Payments.

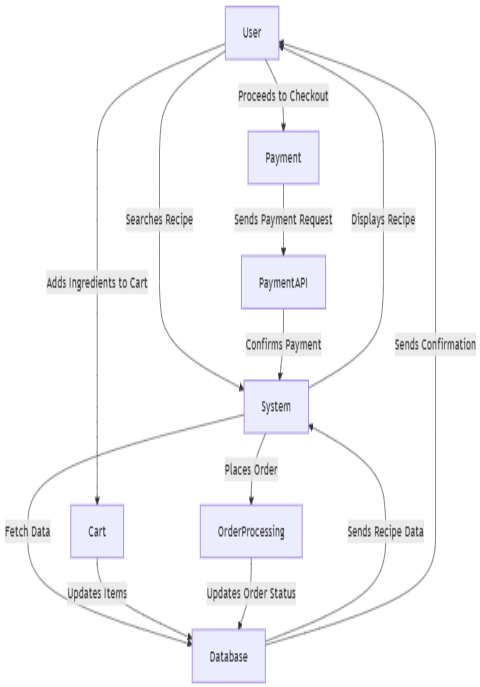
5.2 Logical & Physical Schema

- SQL document-based structure for scalability.

6. Data Flow & System Behavior

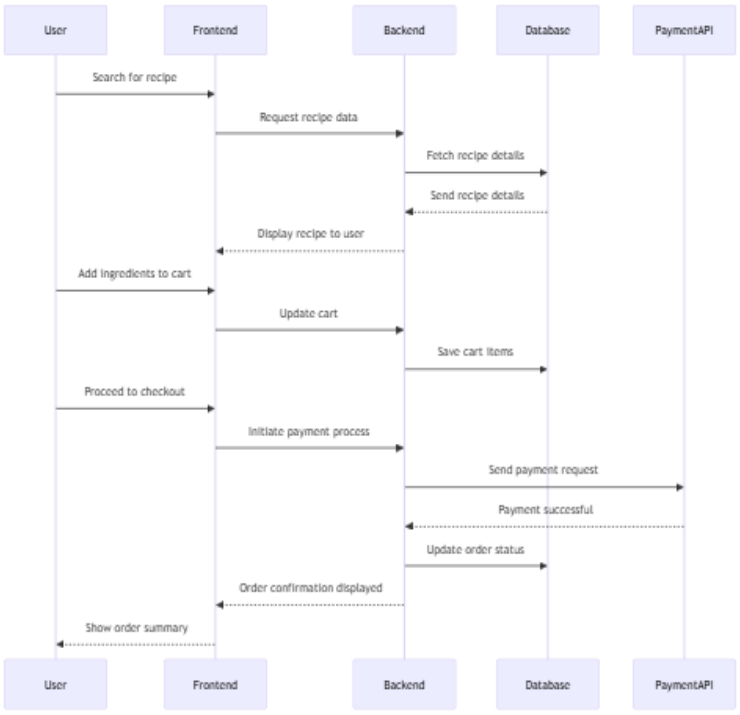
6.1 Data Flow Diagram (DFD)

- **Description:** This diagram represents how data moves through the system. It shows the input (user searches for recipes), processing (fetching data from the database), and output (displaying recipes or processing an order).



6.2 Sequence Diagram

- **Description:** This diagram illustrates the step-by-step interaction between users, the frontend, backend, and database. It details the request flow, such as when a user selects a recipe, adds ingredients to the cart, and completes the payment.

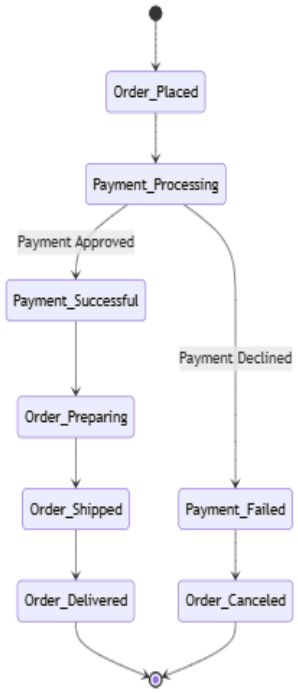


6.3 Activity Diagram

- **Description:** This diagram visualizes different workflows within the application. For example, it outlines the steps of browsing a recipe, adding it to the cart, and completing a transaction

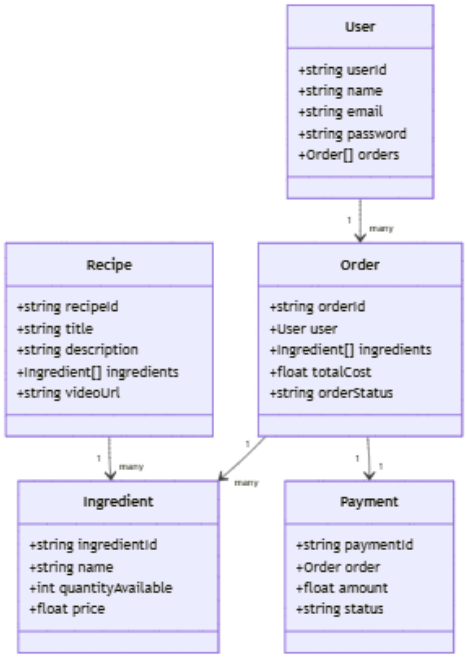
6.4 State Diagram

- **Description:** This diagram represents different states of an object in the system. For example, an order can transition from **Pending → Confirmed → Shipped → Delivered**.



6.5 Class Diagram

- **Description:** This diagram defines the structure of the system by showing classes, attributes, and relationships. Key classes include **User, Recipe, Ingredient, Order, and Payment**.

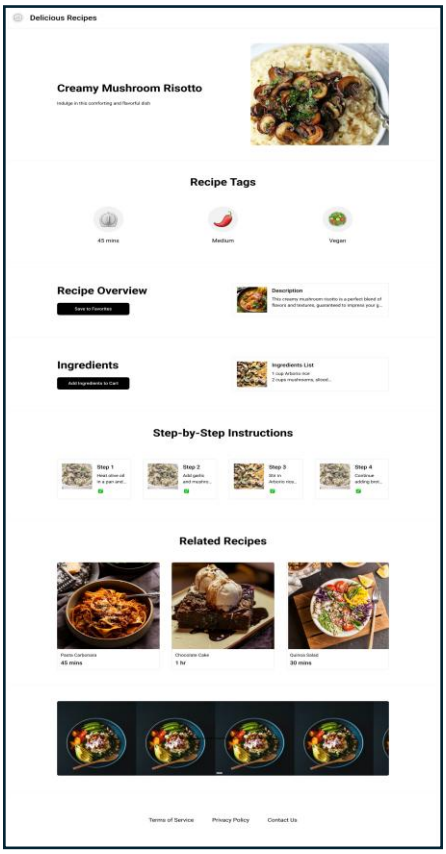


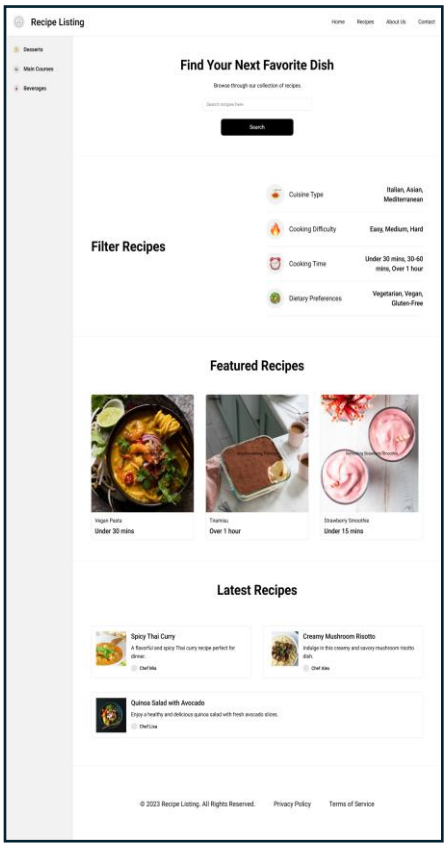
7. UI/UX Design & Prototyping

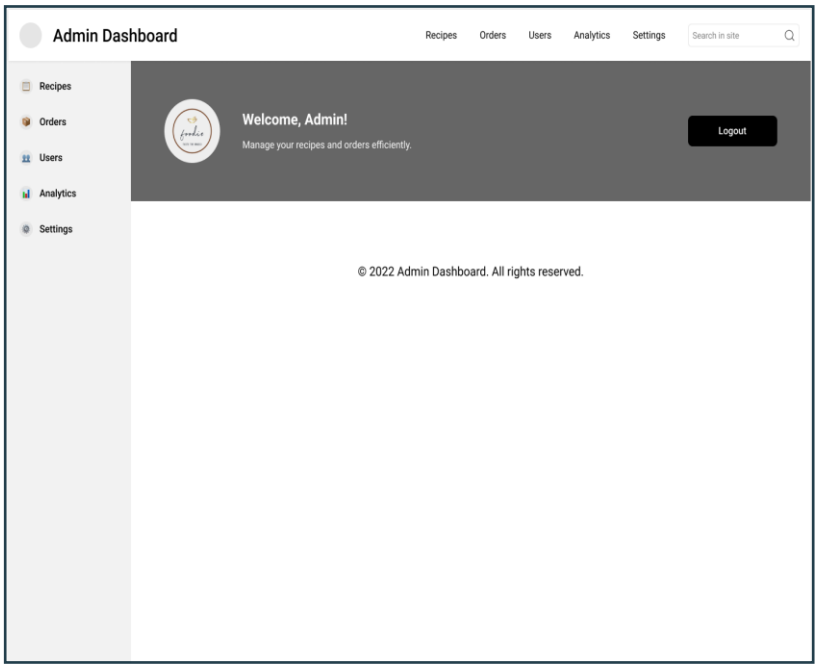
7.1 Wireframes & Mockups

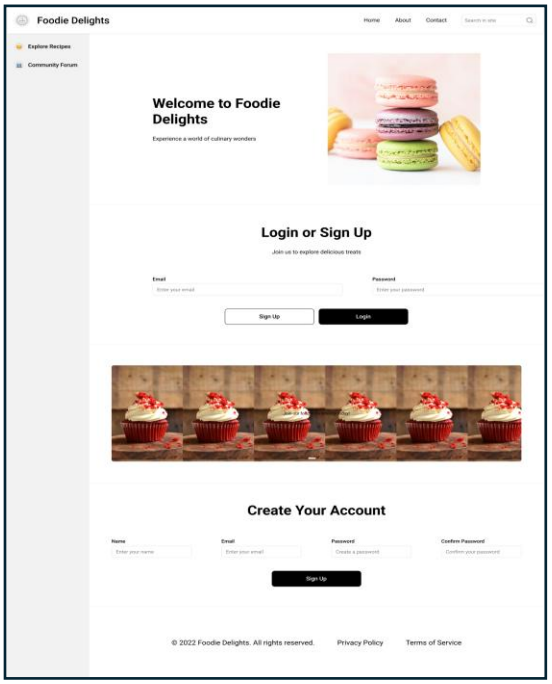
- Screens for homepage, recipe details, order page, and admin dashboard.

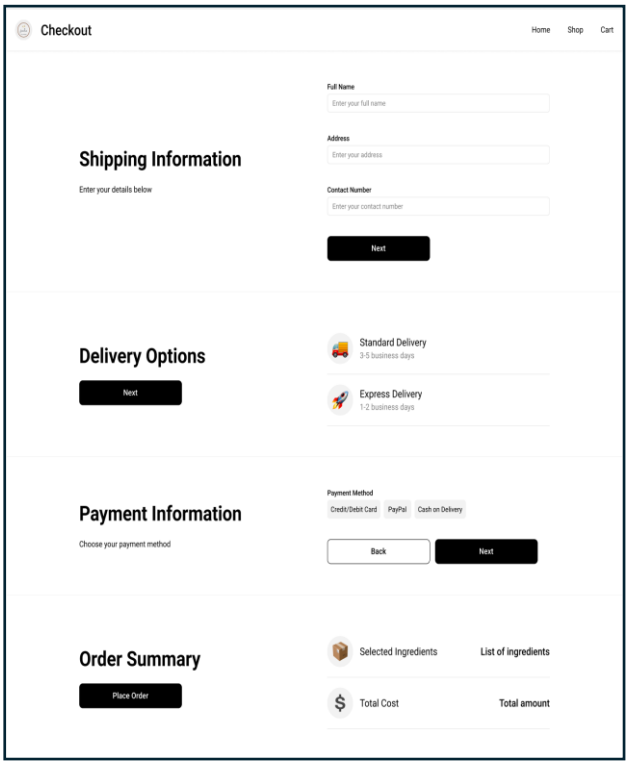
7.2 UI/UX Guidelines

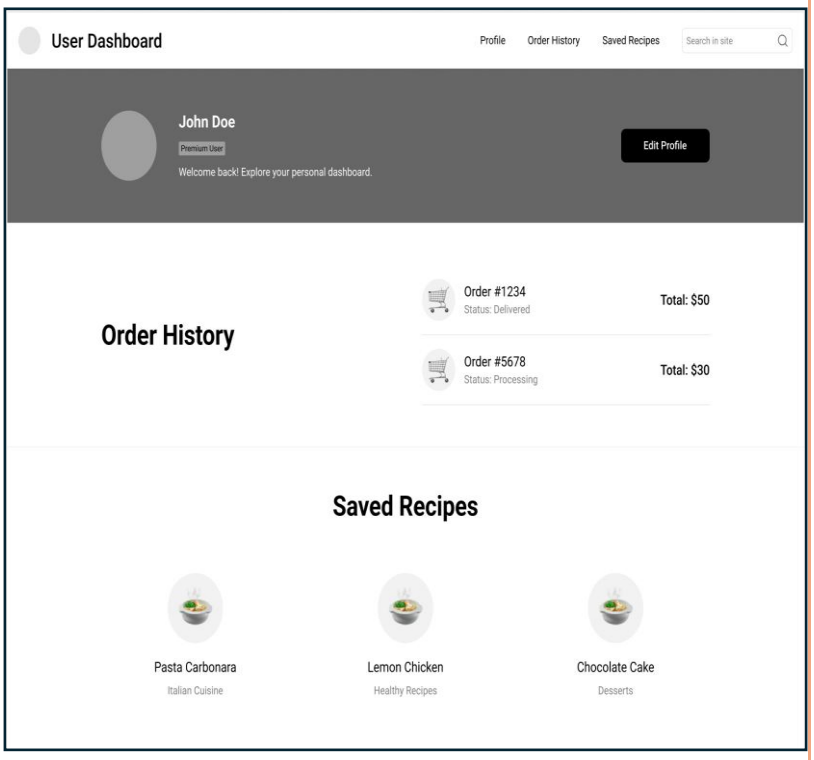












8. System Deployment & Integration

8.1 Technology Stack

- **Frontend:** React.js
- **Backend:** .NET Core API
- **Database:** SQL (MongoDB/Firebase)
- **Hosting:** AWS
- **Payment Gateway:** Stripe/PayPal

8.2 Deployment Diagram & Component Diagram

- Showcasing how the software components interact and are hosted.

9. Implementation (Source Code & Execution)

9.1 Source Code & Version Control

- **Repository:** GitHub for version control.
- **Branching Strategy:** Feature branching with pull requests.
- **CI/CD Integration:** Automated testing and deployment pipeline.

9.2 Deployment & Execution

- **README File:** Includes installation steps, dependencies, and execution guide.

- **Live Demo Link:** Hosted on AWS.

10. Testing & Quality Assurance

10.1 Test Cases & Test Plan

- Functional and usability testing for major features.

10.2 Automated Testing (if applicable)

- Unit tests for API endpoints.

10.3 Bug Reports & Fixes

- Documentation of resolved issues.

11. Final Presentation & Reports

11.1 User Manual

- Guide for users to navigate the platform.

11.2 Technical Documentation

- System architecture and database schema details.

11.3 Project Presentation (PPT/PDF)

- Summary of project implementation and challenges.

11.4 Video Demonstration (Optional)

- A recorded walkthrough showcasing key functionalities.

12. Future Enhancements

1. **AI-Powered Recipe Recommendations** based on user preferences.
2. **Integration with Smart Kitchen Devices** for interactive cooking.
3. **Subscription Model** for exclusive premium recipes.
4. **Group Packages for Friends & Gatherings** to allow users to order larger ingredient bundles tailored for group cooking experiences and special events.

13. Conclusion

The Food Recipe Web Application bridges the gap between **cooking inspiration and ingredient accessibility**. By combining a **blog-like experience with e-commerce functionality**, it provides users with a seamless journey from recipe discovery to meal preparation.