MenuMate Project Report

Contents

1. Introduction and Summary of Project

MenuMate is a robust and user-friendly restaurant management application designed to streamline the process of managing menu items, ingredients, categories, and customer orders. Built with ASP.NET Core MVC, this application provides an intuitive interface to organize menu offerings and track ingredient availability.

Project Objective

The primary goal of MenuMate is to reduce the operational overhead associated with manual menu and inventory management in restaurants. By offering a digital platform for handling ingredients, categories, products (menu items), and orders, MenuMate enhances productivity and minimizes the likelihood of human errors, ultimately improving customer service and inventory control.

Core Functionalities

The application offers comprehensive features that cater to different needs within a restaurant's operations:

- **Ingredient Management**: Allows users to create, update, view, and delete ingredients. Each ingredient can be linked to multiple products, providing a clear overview of what items require specific ingredients.
- **Category Management**: Organizes products by categories (e.g., appetizers, main courses), which improves menu navigation and presentation for both restaurant staff and customers.
- **Product Management**: Each product (menu item) includes information such as name, description, price, stock level, and associated ingredients. Products can be easily added, updated, or deleted as needed.
- **Order Management**: Handles customer orders and tracks inventory changes in real-time, ensuring that ingredient stock levels are always accurate.

MenuMate’s modular design and flexible structure make it scalable for restaurants of various sizes, from small cafes to larger multi-branch establishments.

2. Development Approach

Technology Stack

The following technologies were employed to build MenuMate, each chosen for its compatibility with ASP.NET Core MVC and its ability to handle complex data and user interactions.

1. **ASP.NET Core MVC**: A framework that encourages a Model-View-Controller structure, separating application logic into distinct layers for better code organization and maintainability.

2. **Entity Framework Core**: Used as the Object-Relational Mapping (ORM) tool to handle database interactions. Entity Framework simplifies CRUD operations and provides LINQ-based querying for complex data retrieval.
3. **Bootstrap**: A front-end CSS framework used to create responsive and attractive user interfaces. Bootstrap components are integrated into Razor views to provide a consistent design across pages.
4. **SQL Server**: Chosen as the main database for storing all application data. SQL Server's robustness and compatibility with Entity Framework make it suitable for handling complex relationships between data tables.
5. **ASP.NET Identity**: Used for implementing user authentication and authorization, providing secure login, password hashing, and session management.

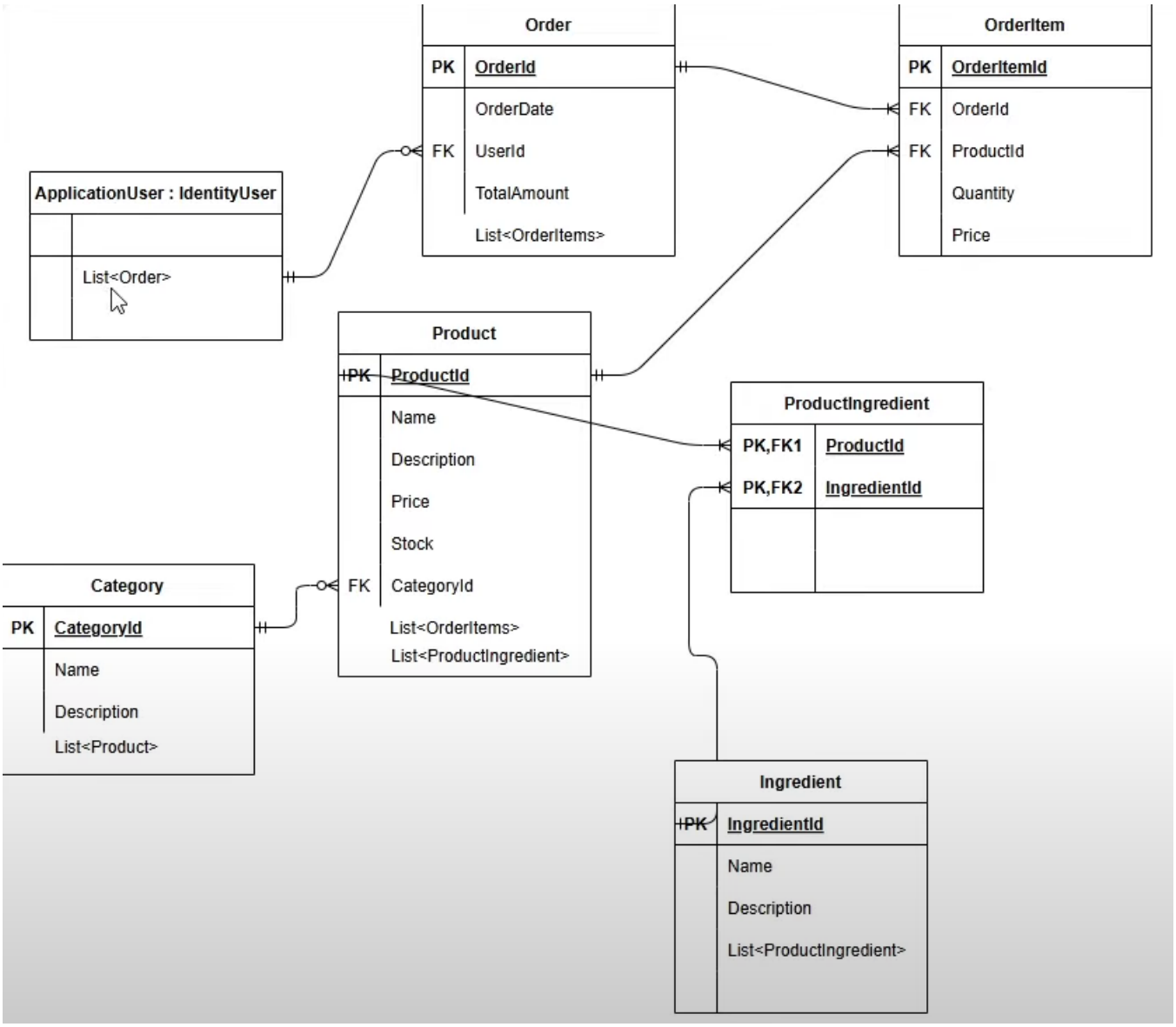
Development Methodology

The team adopted the **Agile methodology** for MenuMate's development, allowing for iterative improvements through regular sprints. Each sprint focused on a specific set of functionalities, followed by team reviews, testing, and refinement. This approach allowed the team to adapt to changing requirements and focus on enhancing key areas of the application progressively.

Key Development Phases

1. **Planning and Requirement Gathering**: This phase involved understanding the primary needs of restaurant managers, including ease of navigation, inventory management, and order processing.
2. **Database Design**: Using Entity Framework's Code-First approach, the team created database models for `Ingredients`, `Categories`, `Products`, and `Orders`. This phase also included defining relationships, such as one-to-many (category to product) and many-to-many (product to ingredient).
3. **Front-End Development**: With Bootstrap and Razor views, the team crafted an intuitive and responsive UI. Focus was placed on creating a seamless user experience with accessible design patterns.
4. **Back-End Development**: Controllers were developed to handle business logic, including CRUD operations, validation, and data processing.
5. **Testing and Bug Fixing**: Testing was carried out in each sprint to identify issues and resolve bugs, ensuring robust performance and functionality.
6. **Deployment Preparation**: The final stages included preparing the application for deployment, refining UI elements, and ensuring compatibility with different browsers.

3. Flowchart of System Components



This flowchart depicts the relationship between controllers, models, and views, illustrating how data flows through the application, from user interaction on the front end to database operations on the back end

4. Role of Team Members

Name	Role	Responsibilities
Nikitha Pashap (24706882)	Frontend Developer	Designed and implemented the user interface using Bootstrap and Razor views. Focused on creating a responsive, user-friendly, and accessible design. Collaborated with the backend team to ensure seamless data integration and UI consistency.
Joy Dhar (24698298)	Backend Developer	Developed core application logic, created and managed controllers, and implemented CRUD operations for ingredients, categories, and products. Worked on server-side validation, business logic, and data handling with Entity Framework to ensure accurate data flow and functionality.

5. Technical Challenges and Solutions

1. **Database Relationships**:
   - **Challenge**: Managing many-to-many relationships between `Products` and `Ingredients`.
   - **Solution**: Used a bridge table `ProductIngredient` with foreign keys to handle the association, allowing each product to have multiple ingredients and vice versa.
2. **Image Management for Products**:
   - **Challenge**: Handling image uploads and storage for product items.
   - **Solution**: Developed a file upload system in the `ProductController` that saves images to a specified directory, storing the file path in the database for retrieval.
3. **User Authentication**:
   - **Challenge**: Ensuring secure user authentication and authorization.
   - **Solution**: Implemented ASP.NET Identity, which provides out-of-the-box security features such as password hashing and token-based authentication.
4. **Session Management**:
   - **Challenge**: Managing user sessions without impacting performance.
   - **Solution**: Configured session settings with optimized timeout periods to balance user convenience and server efficiency.

6. Testing and Quality Assurance

The team conducted comprehensive testing at each stage to ensure that MenuMate meets functionality, usability, and performance standards. Testing types included:

- **Unit Testing**: Focused on individual functions and modules, especially for controllers.
- **Integration Testing**: Verified interactions between components, ensuring smooth data flow between models, views, and controllers.
- **User Acceptance Testing**: Simulated real-world usage to verify ease of navigation, responsiveness, and accuracy in data handling.

Defects were tracked and categorized based on severity, with critical bugs resolved immediately, ensuring high-quality deliverables by project completion.

7. Acknowledgments

The team would like to extend gratitude to the online communities, resources, and official documentation that provided guidance throughout the project. Microsoft's ASP.NET Core documentation and Bootstrap's official guide were invaluable for understanding and implementing various features. Additional support was drawn from Stack Overflow discussions and tutorials on Entity Framework.

8. References

1. **ASP.NET Core Documentation**: A complete guide to ASP.NET Core MVC, covering all foundational concepts and advanced features.
    - ASP.NET Core Documentation
2. **Bootstrap Documentation**: Documentation for Bootstrap components, used to enhance the user interface.
    - Bootstrap Documentation
3. **Entity Framework Core Documentation**: Guide to using Entity Framework Core, essential for database interactions in the project.
    - EF Core Documentation
4. **ASP.NET Identity**
   ASP.NET Identity provides authentication and authorization support for ASP.NET applications. It includes features like password hashing, account confirmation, and session management, essential for securing the MenuMate application.

   Link: ASP.NET Identity Documentation

5. **Agile Methodology**
   Agile development principles and practices, which emphasize iterative development, collaboration, and flexibility. This methodology guided the project's development, enabling incremental improvement through sprints.

Agile Alliance - Agile Development Principles

6. **Designing Data Models for Many-to-Many Relationships**
   This resource covers techniques for setting up many-to-many relationships in databases, such as using a join table. It's especially relevant for MenuMate's relationship between products and ingredients.

   Many-to-Many Relationships in Entity Framework

7. **Security Best Practices in ASP.NET Core**
   A guide to implementing secure coding practices in ASP.NET Core, covering data protection, authentication, and authorization.

   Link: Security Best Practices in ASP.NET Core

8. **Microsoft SQL Server**
   An overview of Microsoft SQL Server, used as the database for storing MenuMate's data. This resource covers setup, querying, and management practices.

   Link: Microsoft SQL Server Documentation