**CS673 Software Engineering**

**Team 1 - Menu Match**

**Software Design Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Dirgha Jivani | Team Leader |  | 12/08/2024 |
| Roshni Dodhi | Requirement Leader |  | 12/08/2024 |
| Prayushi Khandelwal | Security Leader | *P.K* | 11/07/2024 |
| Jiho Cheon | QA Leader | *Jiho Cheon* | 12/08/2024 |
| Pratyush Patel | Design Leader |  | 12/08/2024 |
| Vignesh S | Configuration Leader | *Vignesh S* | 12/08/2024 |
| Anshul Raj | Design and Implementation Leader | **AR** | 11/17/2024 |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_3ipvmjgn6clp)

[Class Diagram](#_ky60nv8suxxm)

[UI Design (if applicable)](#_7ucksmkf6rzx)

[Database Design (if applicable)](#_tcmuor4nl1kz)

[Security Design](#_x18fj36s1121)

[Business Logic and/or Key Algorithms](#_mtfbusfb0eq3)

[Design Patterns](#_9zvwkmc4luo5)

[Any Additional Topics you would like to include.](#_15tmymhipvdv)

[References](#_50ojo9i46ytq)

[Glossary](#_8n34lvocupub)

# Introduction

The project is motivated by the challenges faced by international students and others who follow specific diets such as vegan, gluten-free, or halal. Traditional restaurant review platforms often lack the nuanced information necessary for such users, making dining choices difficult. This document outlines the design goals, architectural layout, and functional aspects of the platform, aiming to bridge that gap by providing detailed, customizable reviews that focus on dietary needs. Users will be able to filter reviews not only based on general restaurant quality but also on how well a restaurant caters to their specific dietary preferences. This approach will empower users to make more informed choices that align with their unique needs.

The design goals of this software system are as follows:

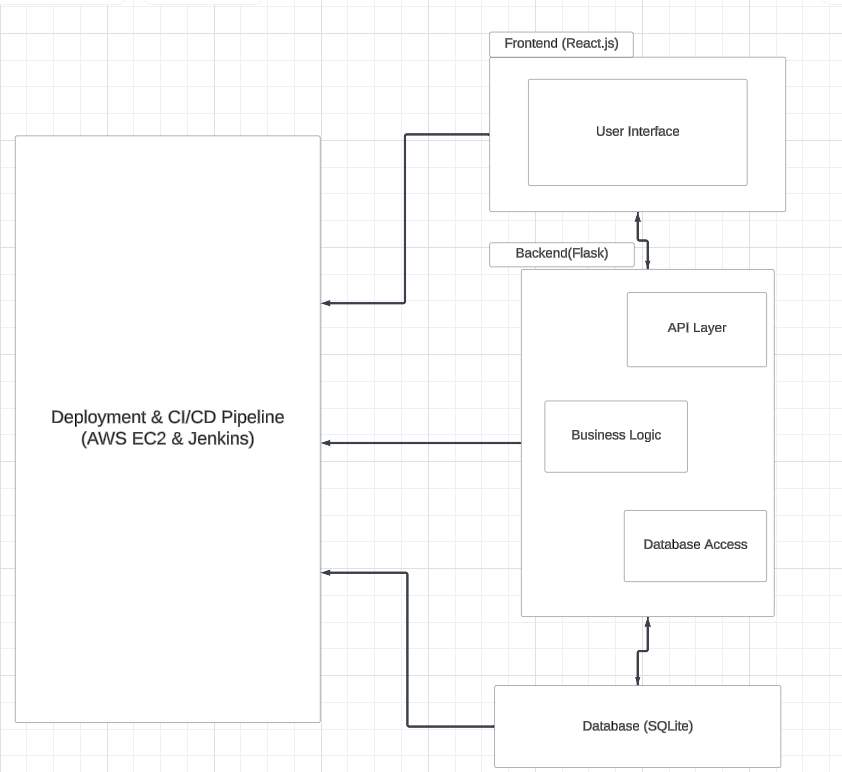
1. **Enhanced User Experience**: Deliver a user-friendly interface that allows users to easily filter and find restaurants that meet their dietary requirements.
2. **Personalized Recommendations**: Enable tailored suggestions and ratings based on specific dietary categories, offering greater insight into the suitability of menu items.
3. **Scalability and Performance**: Ensure that the system can handle increasing user traffic and expand to other regions over time.
4. **Accurate and Relevant Data**: Provide a reliable database of restaurant information, constantly updated to reflect current menu options and reviews specific to dietary preferences.

# Software Architecture

The software system is composed of several interrelated components that work together to provide the full functionality of Menu Match.

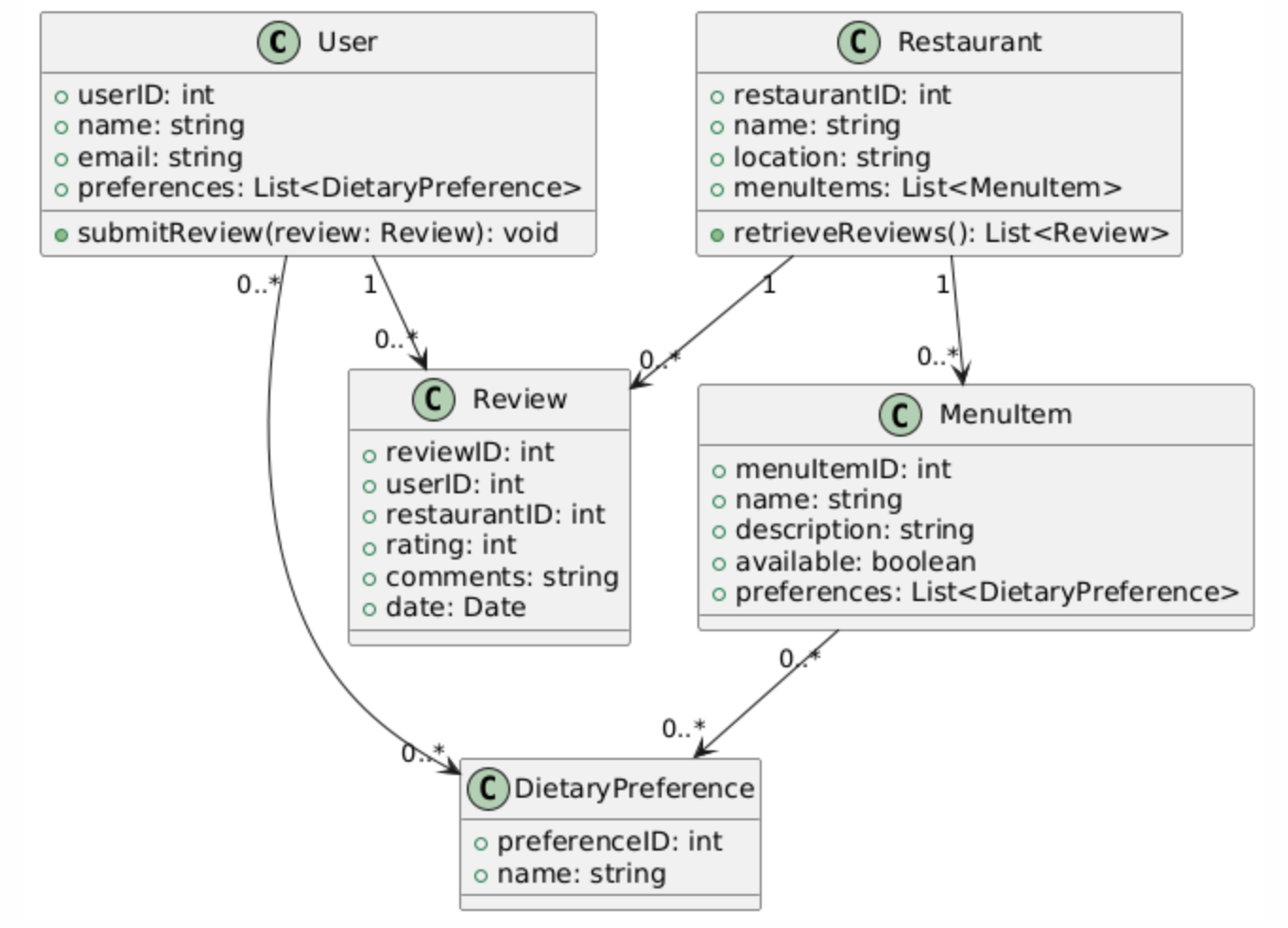
**Primary Components**:

* **Frontend (React Framework)**: This component is responsible for all user interactions, including searching for restaurants and submitting reviews.
* **Backend (Node.js)**: Handles API requests, authentication, and database operations.
* **Database (MySQL)**: Stores structured data including user profiles, restaurant details, reviews, and dietary preferences.
* **CI/CD Pipeline (Jenkins)**: Automates the development, testing, and deployment processes.
* **Deployment (AWS)**: The platform is deployed on AWS EC2 to ensure reliability and scalability.



# Class Diagram

This section describes the classes and their relationships, representing the core entities of the system, such as users, restaurants, reviews, and dietary preferences.

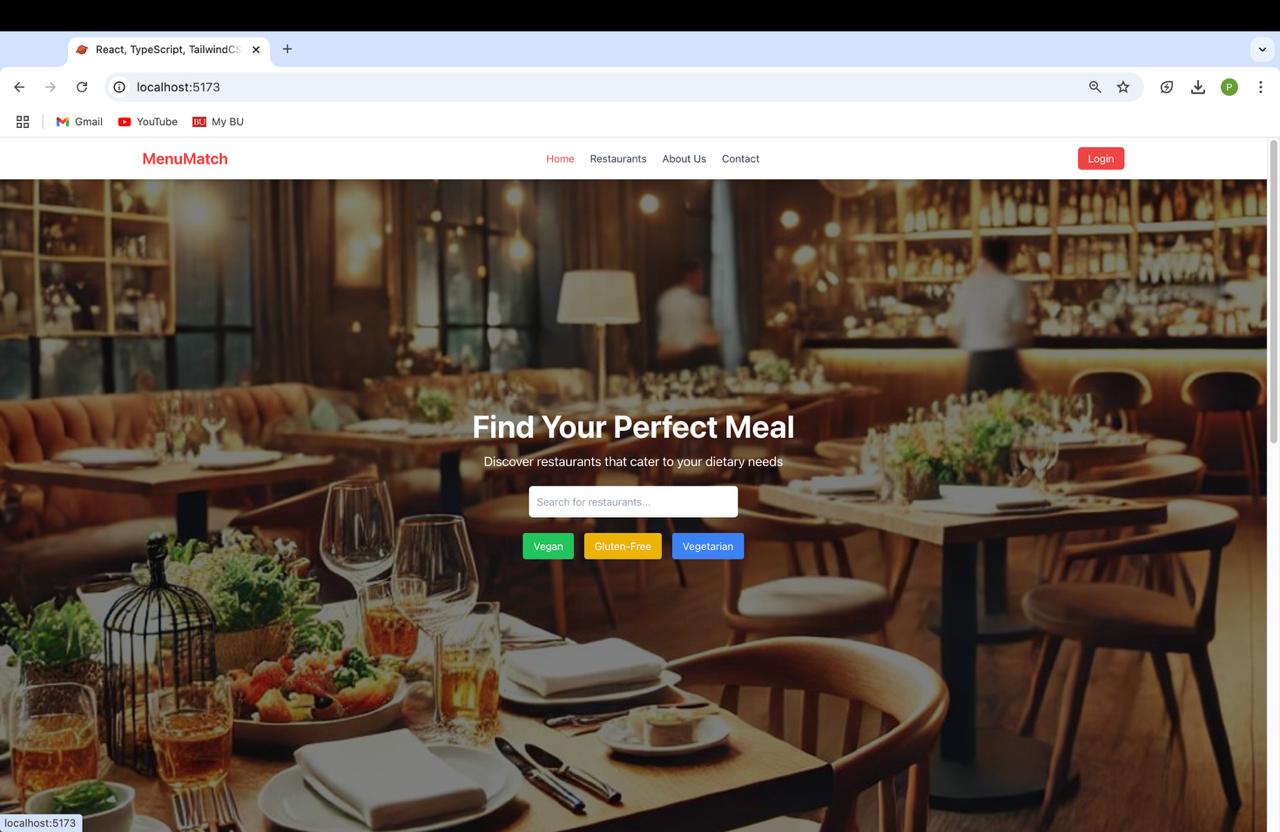


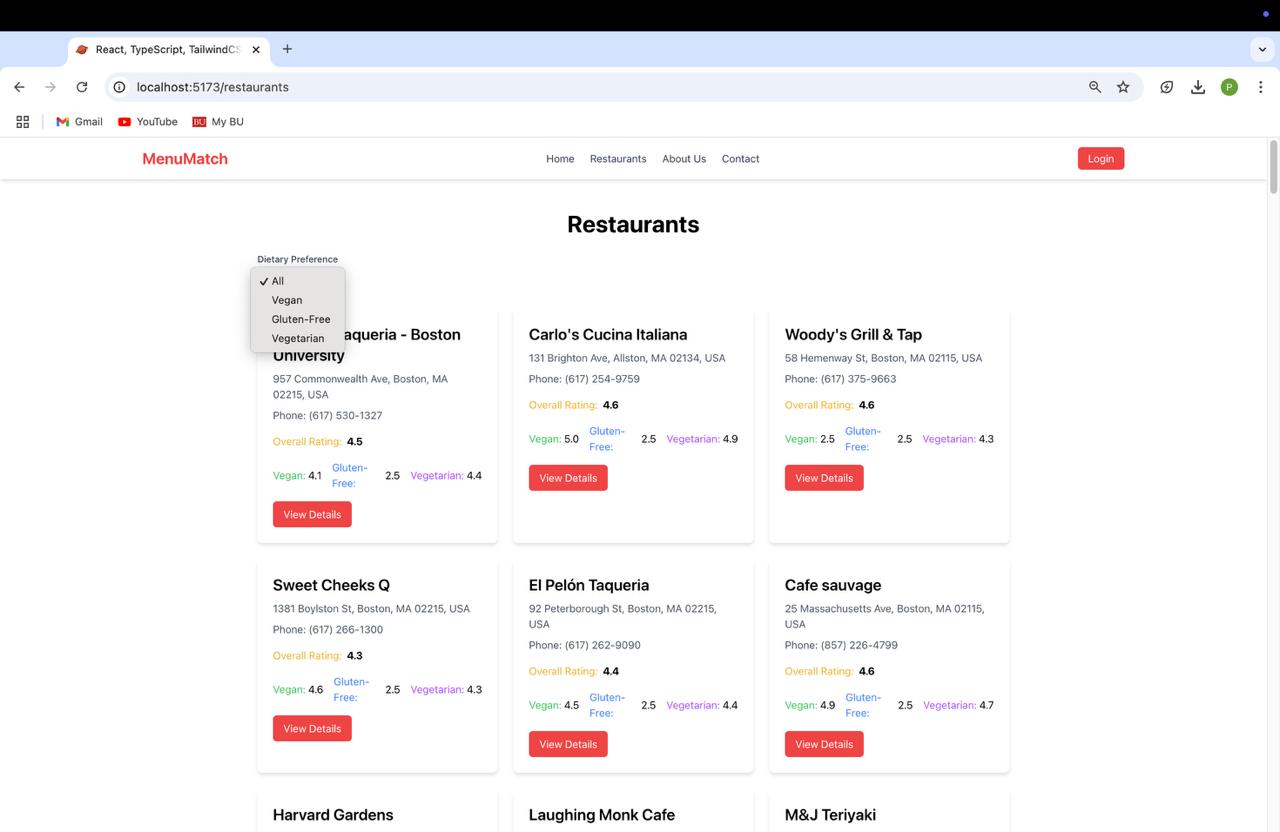
**Explanation**:

* **User**: Represents individuals interacting with the platform, who can submit reviews and have specific dietary preferences.
* **Restaurant**: Contains information about restaurants and their menu items.
* **Review**: Represents feedback provided by users about restaurants.
* **DietaryPreference**: Links dietary preferences to both users and menu items.

# UI Design (if applicable)

Example welcome page & sign-in page for web





The user interface of the Menu Match platform is designed using the React framework. The focus is on simplicity and ease of use. The major components include:

* **Search Feature**: Enables users to find restaurants based on their dietary preferences.
* **Restaurant Listings**: Displays a filtered list of restaurants based on user-defined dietary criteria.
* **Review Submission Form**: Allows users to provide feedback and rate restaurants based on their experiences.

**Search Bar**

* Developed a dynamic, **filterable search bar** for users to find restaurants based on food type, dietary preferences, and other criteria.
* Implemented real-time filtering and state management to provide an interactive and seamless experience.

**Restaurant Cards**

* Designed reusable **restaurant cards** to display key information such as name, ratings, cuisine type, and distance.
* Integrated **conditional rendering** to dynamically display restaurant data fetched from the backend.

**Responsive Design**

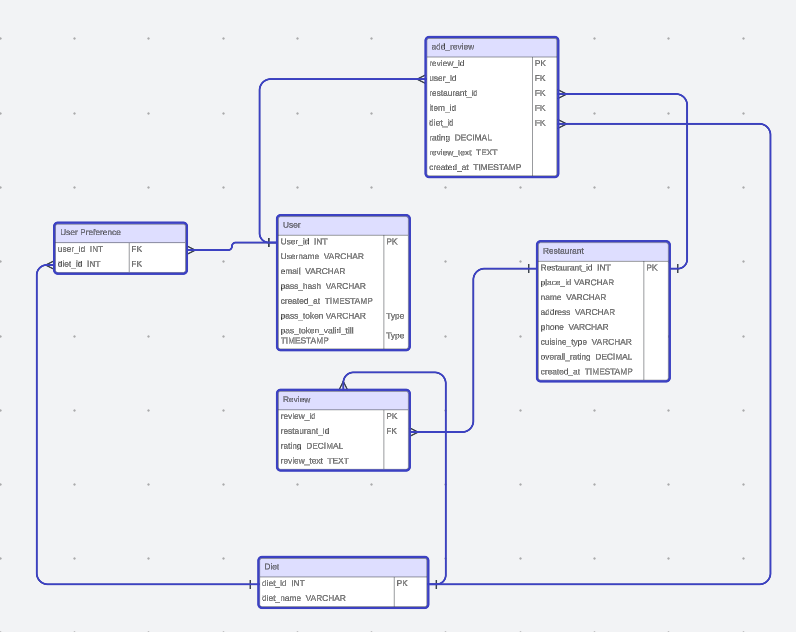
* Used **React** for dynamic component rendering and **Tailwind CSS** for responsive, mobile-first design, ensuring the app is accessible across all devices.
* Ensured the UI adapts fluidly to various screen sizes, offering an optimal user experience on both mobile and desktop.

**State Management**

* Implemented **Reactjs state management** to handle user inputs (e.g., search filters) and data fetched from APIs (e.g., restaurant information).
* Ensured efficient data flow and real-time updates without page reloads.

# Database Design (if applicable)

The database is designed using SQLite and serves as the backbone for data storage. The system stores users, restaurants, reviews, and menu items in relational tables.



# Security Design

To ensure security, several mechanisms will be place:

* **Containerization**: The system runs inside Docker containers, providing isolated environments for each service.
* **HTTPS**: All communication between clients and the server is encrypted, ensuring secure data transmission.
* **Authentication**: Users are authenticated using secure tokens (JWT), ensuring only authorized users can access certain features.
* **Data Encryption**: Sensitive data is encrypted both when stored and during transmission.

# Business Logic and/or Key Algorithms

The primary business logic revolves around providing personalized restaurant recommendations based on the user’s dietary preferences. The recommendation engine uses an algorithm that matches users’ preferences with available restaurant data.

# Design Patterns

The platform employs several design patterns to streamline development and maintainability:

* **Model-View-Controller (MVC)**: Separates the user interface (View), application logic (Controller), and data storage (Model), making it easier to manage and extend the application.
* **Factory Pattern**: Used to instantiate different types of user roles, such as standard users and restaurant owners, based on their profile.

# Rest APIs

The backend exposes RESTful APIs that allow the frontend to interact with the system. The primary endpoints include:

* **/api/restaurants**: Retrieves a list of restaurants filtered by dietary preferences.
* **/api/reviews**: Allows users to submit reviews and view existing ones for specific restaurants.

# Any Additional Topics you would like to include.

# References

# Glossary