**Software Requirements Specification (SRS) for RemixMeals**

**1. Introduction**

**1.1 Document Purpose**

This Software Requirements Specification (SRS) defines the detailed functional and non-functional requirements for the "RemixMeals" platform. The SRS is intended for software developers, UI/UX designers, testers, and stakeholders, providing a comprehensive reference to guide the design, implementation, and validation of the system. It ensures that the development team has a clear and complete understanding of the project requirements, minimizing ambiguities and ensuring alignment with stakeholder expectations.

**1.2 Product Scope**

RemixMeals is a web-based application that assists users in creatively utilizing leftover ingredients to generate new recipe ideas from a database, aiming to minimize food waste and promote sustainable living. The platform will require users to register and log in to access the recipe suggestion features. By offering practical and creative solutions, RemixMeals seeks to empower individuals to reduce food waste, save money, and explore new culinary possibilities.

**1.3 Document Overview**

* Section 2: Overall system description including product perspective, main functionalities, user characteristics, constraints, and dependencies.
* Section 3: Specific detailed requirements including functional and non-functional specifications.
* Section 4: Supporting information including use cases, architecture overview, and references.

**1.4 Definitions, Acronyms, and Abbreviations**

* UI: User Interface
* API: Application Programming Interface
* DB: Database
* SPA: Single Page Application
* HTTPS: HyperText Transfer Protocol Secure

**2. Overall Description**

**2.1 Product Perspective**

RemixMeals is an independent web application that provides a novel solution to food waste by helping users creatively reuse their leftover ingredients. It connects to a database containing a wide variety of recipes. Hosted on a cloud environment, the application will leverage scalable infrastructure to ensure high availability and reliability.

**2.2 Product Functions**

* User registration and login functionality to access recipe generation features.
* Ingredient input field allowing users to enter and submit a list of available ingredients after login.
* Real-time recipe suggestion display, including recipe title, list of required ingredients, cooking instructions, and estimated preparation time.
* Option for users to refresh and regenerate recipe suggestions to explore more options.
* Feedback mechanism enabling users to rate the usefulness of suggested recipes, facilitating continuous database updates.

**2.3 User Characteristics**

* Primary Users: Home cooks, students, busy professionals, and eco-conscious individuals who are interested in minimizing food waste and exploring creative cooking.
* Technical Proficiency: Users are expected to have basic to moderate familiarity with web applications. The interface must be intuitive and straightforward, requiring minimal instructions.
* User Roles: The system distinguishes between regular users and a single admin user. The admin is responsible for managing users and recipe entries. By default, the first registered user is designated as the admin.

**2.4 Constraints**

* Application must be compatible with the latest versions of Chrome, Firefox, and Safari browsers on desktop and mobile devices.
* Only registered users can access recipe generation features; guest users have no access.
* Recipe suggestions depend on the breadth and depth of the database content.
* Web application must comply with WCAG 2.1 AA accessibility standards to ensure usability for individuals with disabilities.

**2.5 Assumptions and Dependencies**

* The application will primarily use internal datasets stored in a database.
* Real-time communication between frontend and backend will use socket programming.

**3. Overall Description**

**3.1 External Interfaces**

**3.1.1 User Interfaces**

* Responsive single-page web application (SPA) supporting both desktop and mobile views.
* Registration page allowing users to sign up and login securely.
* Clean and simple homepage featuring an ingredient input field, search button, Favorites and random recipe.
* Ingredient input with auto-suggest dropdown for common items.
* Visual recipe cards displaying the recipe name, ingredient list, preparation steps.

**3.1.2 Hardware Interfaces**

* End-user devices such as desktops, laptops, tablets, and smartphones equipped with modern browsers.

**3.1.3 Software Interfaces**

* RESTful API integration for internal database operations.
* Database connectivity using MongoDB.
* Use of Materialize CSS framework for frontend styling.

**3.1.4 Communication Interfaces**

* All communication between client and server will use HTTPS to ensure secure data transmission.

**3.2 Functional Requirements**

**3.2.1 User registration and authentication**

* FR1: The system shall allow users to register with an email address and password.
* FR2: The system shall authenticate users securely and allow login/logout functionality.
* FR3: Only authenticated users shall be able to access the Favorite button and can Favorite any recipe.

**3.2.2 Ingredient Input**

* FR4: The system shall provide an input field allowing multiple ingredients to be typed.
* FR5: The system shall validate the ingredients against a predefined database and suggest corrections if needed.

**3.2.3 Recipe Suggestion**

* FR6: Upon submitting ingredients, the system shall query the database and generate relevant recipe suggestions.
* FR7: The system shall prioritize recipe suggestions based on the number of matching ingredients and overall feasibility.

**3.2.4 Recipe Display**

* FR8: Each recipe card shall display the recipe title, ingredient list, preparation instructions, and estimated preparation time.

**3.2.5 Error Handling**

* FR9: If no matching recipes are found, the system shall display a friendly error message suggesting broader or different ingredient entries.

**3.2.6 Admin Features**

* FR10: The system shall assign admin privileges to the first registered user.
* FR11: The system shall allow the admin to create, edit, or delete user accounts.
* FR12: The system shall allow the admin to create, edit, or delete recipes in the database.
* FR13: The system shall display a dashboard to the admin showing the total number of users and recipes currently stored.
* FR14: The system shall restrict access to admin functionality for non-admin users.

**3.3 Non-Functional Requirements**

**3.3.1 Performance Requirements**

* NFR1: Recipe suggestions must be generated and displayed within 20 seconds after user input submission.
* NFR2: The platform must efficiently handle at least 50 concurrent active users without degradation in performance.

**3.3.2 Security Requirements**

* NFR3: All server communications must be encrypted using HTTPS with valid SSL certificates.
* NFR4: Passwords must be securely hashed and stored in the database.
* NFR5: No personally identifiable information (PII) will be shared without consent.

**3.3.3 Usability Requirements**

* NFR6: Users must be able to register, login, input ingredients, and receive recipe suggestions without needing a tutorial.

**3.3.4 Availability Requirements**

* NFR7: The system must maintain an uptime of 99.5% or higher on a monthly basis.

**3.3.5 Maintainability Requirements**

* NFR8: The application architecture must be modular and scalable, facilitating future enhancements.

**4. Supporting Information**

**4.1 System Architecture Overview**

* Frontend: Developed using React.js Single Page Application framework.
* Backend: Implemented using Node.js to manage database queries and user authentication.
* Database: MongoDB used to store static recipe data, user accounts, and feedback logs.
* Frontend Styling: Materialize CSS framework for responsive and clean design.
* The backend will include role-based access control (RBAC) logic to ensure only the admin user has access to sensitive management endpoints.

**4.2 Use Case Descriptions**

**Use Case 1:** User Registration and Login

* Actor: New or Returning User
* Pre-condition: User is on the Registration/Login Page.
* Flow:
  1. User enters username and password.
  2. System validates credentials.
  3. User gains access to the ingredients input and recipe suggestion features.

**Use Case 2:** Ingredient Input and Recipe Suggestion

* Actor: Registered User
* Pre-condition: User is logged in
* Flow:
  1. User inputs available ingredients.
  2. User clicks “Get Recipes”.
  3. System validates ingredients and displays recipes based on validated input.

**Use Case 3:** Regenerate Recipe Suggestions

* Actor: Registered User
* Pre-condition: Recipes are already shown
* Flow:
  1. User clicks “Regenerate” button.
  2. System fetches and displays a new set of recipe suggestions.

**Use Case 4:** Admin Management

* Actor: Admin User
* Pre-condition: Admin is logged into the system
* Flow:
  1. Admin accesses the admin dashboard.
  2. Admin can view total counts of user and recipes.
  3. Admin can select options to create, update, or delete users or recipes.
  4. System performs the requested operation and returns a success or error message.

**Member Contributions to the SRS**

* **Dhananjay Suhas Choudhari (S223482522):** Drafted Introduction, Product Scope, Overall Description.
* **Likith Somashekar (S223602808):** Developed Specific Requirements (Functional and Non-Functional Requirements).
* **Utkarsh Aggarwal (S223274728):** Documented Use Case Descriptions, Admin Features, System Architecture Overview, User Characteristics