EATS SYSTEM
REQUIREMENTS DOCUMENT

CHEFS
GIO, REIHAN, SURAJ

Table of Contents

Introduction

Describe the purpose of this requirements document and outline what it contains.

The purpose of this document is to define the functional, and behavioral requirements for our Eats app. It serves as a blueprint for our team to ensure that the (hypothetical) software implementation aligns with the project's goal. This document will act as the reference for validating system logic, database structure, and UI interactions.

This document also contains:

- Project Overview: Background, rationale, and goals of the recipe app
- Functional Requirements: Core features such as recipe search, shopping list generation, and sharing capabilities
- Non-Functional Requirements: Performance, usability, scalability, and security expectations
- Input and Output Specifications: Details on how users interact with the system and what results they receive
- Process Descriptions: How the system handles recipe storage, retrieval, and shopping list creation
- Constraints and Assumptions: Limitations, dependencies, and conditions affecting development
- Acceptance Criteria: Standards for validating that requirements are met
- References: Supporting materials, definitions, or external references

Description Model

Using text, describe the requirements for your system. Expand on the function section from your project plan. Include requirements for the following categories: Output, Input, Processes, Performance and Security.

1. Input Requirements

The system must accept and validate the following data from users:

- **User Credentials:** Username:3–20 characters, alphanumeric, allowing underscores, password: 8–12 characters, At least one uppercase letter, one lowercase letter, one

number, and one special character, should not contain the username, the user's real name, and email address for registration and login authentication.
- **Search Queries:** Alphanumeric text strings for finding recipes by title, tag, or ingredient.
- **Recipe Data:** Structured data for creating new recipes, including:
    - **Text (Mandatory):** Title, description, step-by-step instructions, preparation time, and serving size.
    - **Structured Lists (Mandatory):** Ingredients with specific quantities and units.
    - **Media (Optional but recommended):** Image files (JPG/PNG) representing the dish. Up to 100 mb of media.
- **Interactions:**
    - **Ratings:** Integer values (1–5) for recipe quality.
    - **Reviews:** Textual feedback comments.
    - **Toggles:** Binary inputs for "Like/Unlike" status and "Add to Shopping List" selection.

2. Output Requirements

The system must generate and display the following information:

- **Curated "For You" Feed:** A dynamic list of recipe cards tailored to user preferences, displaying thumbnail images, titles, and aggregate star ratings.
- **Recipe Details:** A full-page render of a recipe, including high-resolution images, formatted ingredient lists, and sequential instruction steps.
- **Search Results:** A filtered list of recipes matching user queries, with visual indicators for "No Results Found" if applicable.
- **Shopping List:** An aggregated view of ingredients selected by the user, persisted across sessions.
- **Shareable Links:** Unique URLs generated for specific recipes to be shared via external apps (SMS, Social Media).
- **System Notifications:** Toast messages or alerts for confirmation ("Recipe Uploaded," "Added to Cart") or error handling ("Network Error," "Profanity Detected").

3. Process Requirements

The system must perform the following internal logic and calculations:

- **Recommendation Algorithm:** Analyze user history (likes, views) to rank and display relevant content on the "For You" page.
- **Input Validation & Sanitization:**
    - Verify that mandatory fields (Title, Ingredients) are present during recipe upload.
    - **Profanity Filter:** Scan review text against a banned word list before committing to the database.

- **Aggregations:** Automatically recalculate the average star rating of a recipe whenever a new rating is submitted.
- **Shopping List Management:** Parse selected ingredients from a recipe and append them to the user's persistent shopping list table.
- **Image Optimization:** Resize or compress uploaded recipe images to ensure optimal loading speeds without sacrificing quality.

4. Performance Requirements

- **Latency:** The "For You" feed and Search Results must load within **2 seconds** under normal network conditions to ensure user retention.
- **Concurrency:** The system must support multiple users uploading recipes and writing reviews simultaneously without data collision or database deadlocks.
- **Availability:** The application requires 99.9% uptime, particularly for the read-heavy operations (browsing and searching).
- **Scalability:** The database must be structured to handle a growing library of high-resolution images and recipe entries without degrading search query performance.

5. Security Requirements

- **Authentication:** All write actions (Uploading, Reviewing, Liking) require a valid, authenticated user session.
- **Data Protection:** User passwords must be hashed and salted before storage.
- **Content Security:**
    - Prevent SQL injection via search bars and input fields.
    - Ensure file uploads are restricted to image formats (prevent executable file uploads).
- **Privacy:** User specific lists (Shopping List, Liked Recipes) must be accessible only by the owner of that account.

Class Diagram

Create a class diagram. The Class Diagram should contain all of the system objects, their attributes, and any known methods. This diagram may be included as a separate file – it does not need to be inserted into this Word document.

ClassDiagramProjecUpdated.vsdx

Use Case Diagram

Create a Use Case Diagram for all of the "uses" of your system. This diagram may be included as a separate file – it does not need to be inserted into this Word document.

UseCaseDiagramProject.vsdx

Use Case Scenarios

Create a full description Use Case Scenario (detailed descriptions) for each use case of the system. This full scenario should include an enumerated list of steps involved in the activity as well as any exception conditions.

System Sequence Charts

For each Use Case Scenario, provide a sequence diagram. Use your class diagram, use case diagram and scenarios to create the corresponding System Sequence Diagram.