Recipe Generator Project Document

1. Project Overview

- **Purpose**: To create a recipe generator that allows users to input ingredients and receive a list of recipes that include relevant ingredients and detailed instructions.
- **Platform**: Web and mobile app (Android and iOS).
- **Technology Stack**:
    - **Frontend**: HTML, CSS, JavaScript (React/Flutter for mobile app).
    - **Backend**: Node.js with Express or Python with Flask/Django.
    - **Database**: Firebase, MongoDB, or PostgreSQL.

2. Core Features

2.1. User Input

- **Ingredient Search**:
    - Users can **type ingredients** into a search bar.
    - Autocomplete suggestions for commonly used ingredients.
    - Option to add or remove ingredients from the list.
    - Allow input validation (e.g., restrict special characters or ensure items are correctly spelled).

2.2. Recipe Search and Results

- **Recipe List**:
    - Display a list of recipes based on the entered ingredients.
    - Each recipe in the list shows a **recipe name** and a **short description**.
- **Detailed Recipe View**:
    - When a user selects a recipe, they will see:
        - **Ingredients**: Full list of ingredients with quantities.
        - **Instructions**: Step-by-step instructions for preparation.
    - Optional: **Save recipe** to a favorites list or a personal collection.

3. User Interactions

3.1. Entering Ingredients

- Users begin by typing in ingredients into a **search bar**.

- Autocomplete helps users find common ingredients faster.
- Users can add multiple ingredients to their search.
- Once done, they can press **Search** to find recipes.

3.2. Viewing Recipes

- The **recipe list** is displayed based on ingredients entered.
- Users select a recipe to view its details, including ingredients and instructions.

4. Technical Requirements

4.1. Frontend

- **Form Validation**: Input field for ingredient entry with validation to handle spelling or common ingredient names.
- **Autocomplete Feature**: Autocomplete dropdown to help users find ingredient names.

4.2. Backend

- **Database**: Store recipes with fields for recipe_name, ingredients, instructions.
- **API Endpoints**:
    - POST /searchRecipes: Accepts a list of ingredients and returns matching recipes.
    - GET /recipeDetails/:id: Returns full details (ingredients, instructions) for a specific recipe.

4.3. Database

- **Structure**:
    - **Recipes Collection/Table**:
        - recipe_id: Unique identifier for each recipe.
        - recipe_name: Name of the recipe.
        - ingredients: List of ingredients needed.
        - instructions: Steps to make the recipe.
    - **Ingredient Collection/Table** (Optional): For managing autocomplete options for ingredients.