CookBook: Your Virtual Kitchen Assistant

1. **Introduction**

**Project Title :** **CookBook: Your Virtual Kitchen Assistant**

**Team ID : NM2025TMID41684**

**Team Leader : MADHAVAN.R**

**Mail id : mr.maddy3009@gmail.com**

**Team Members:**

1. **MADHAVAN.R** – Lead Developer (Frontend & State Management)

**2. LINGESH.ST** – UI/UX Designer & Frontend Developer

**3. LOGESWARAN.P** – Testing & Quality Assurance

**4. MOHAMMED ABIJEETH.M** – Documentation & Deployment Support

1. **Project Overview**

**Purpose:**  
The **CookBook: Your Virtual Kitchen Assistant** application is designed to help users plan, organize, and enhance their cooking experience. It provides an intuitive platform to store and manage recipes, generate shopping lists, and guide users through step-by-step cooking instructions. With features like personalized recommendations, dietary filters, and meal planning, CookBook makes home cooking more accessible, efficient, and enjoyable for beginners and experienced cooks alike.

**Features:**

* **Recipe Management** – Add, edit, and organize personal or shared recipes.
* **Smart Search & Filters** – Find recipes based on ingredients, cuisine, cooking time, or dietary needs.
* **Step-by-Step Cooking Mode** – Guided instructions with timers and tips for each step.
* **Shopping List Generator** – Automatically create and manage shopping lists from selected recipes.
* **Meal Planning** – Plan daily or weekly meals and track nutritional value.
* **Favorites & Recommendations** – Save favorite recipes and receive personalized suggestions.
* **Offline Access** – Access saved recipes without an internet connection.
* **Responsive Design** – Works seamlessly on mobile, tablet, and desktop.

1. **Architecture**

**Component Structure:**  
The application is structured with modular React components such as **Recipe, Search & Filters, ShoppingList, MealPlanner, CookingMode, UserProfile, and NavBar**. Each module handles a specific responsibility and interacts via props and context.

**State Management:**

* **Global State**: Managed through Context API (RecipeContext, ShoppingListContext, MealPlannerContext, UserContext).
* **Persistence**: Achieved with a custom hook (useLocalStorage) for saving recipes, preferences, and shopping lists.
* **Local State**: Controlled using React hooks (useState, useEffect) for UI interactions like search filters, cooking timers, and form inputs.

**Routing:**  
Basic routing is implemented using **React Router**, including a NotFound page for invalid routes.

1. **Setup Instructions**

**Prerequisites:**

* Node.js
* npm

**Installation Steps:**

1. Clone the repository:
2. git clone https://github.com/MADHAVAN-R-bot/CookBook.git
3. Navigate to the project folder:
4. cd MADHAVAN-R-bot/CookBook
5. Install dependencies:
6. npm install
7. Start the development server:
8. npm start
9. **Folder Structure**

**Client (React App):**

**src/components – Reusable UI components (RecipeCard, RecipeForm, NavBar, etc.)**

**src/pages– Pages such as RecipeList, RecipeDetail, AddRecipe, Favorites**

**src/context – Global state providers (if used)**

**src/hooks– Custom hooks (if any)**

**public/– Static assets and base HTML**

**tailwind.config.js (or CSS files if another styling system used)**

1. **Running the Application**

To run the frontend locally:

npm start

The app will be available at [React App](http://localhost:3000/) (http://localhost:3000)

1. **Component Documentation**

**Key Components:**

**RecipeList – Displays all available recipes**

**RecipeDetail – Shows recipe ingredients, steps, and image**

**AddRecipe – Form to add new recipes**

**Favorites – Displays recipes saved as favorites**

**NavBar – Navigation between sections**

**Reusable Components:**  
RecipeCard, IngredientItem, and ShoppingListItem are reusable across different contexts.

1. **State Management**

* **Global State:** Managed using **Context API** (RecipeContext, ShoppingListContext, MealPlannerContext, UserContext).
* **Local State:** Controlled with **React hooks** (useState, useEffect) for handling search filters, timers, cooking steps, and form inputs).
* **Persistence:** Implemented with a custom hook (**useLocalStorage**) to save recipes, shopping lists, meal plans, and user preferences.

1. **User Interface**

The UI is designed to be **intuitive, responsive, and visually engaging**, built with **TailwindCSS** for consistency across devices. Key screens include:

* **Home Dashboard** – Quick access to featured recipes, favorites, and meal planner.
* **Recipe Catalog** – Displays recipes with filtering and search options.
* **Recipe Details** – Step-by-step instructions, ingredients, and nutritional info.
* **Shopping List** – Editable grocery list with item check-off functionality.
* **Meal Planner** – Calendar-style interface for scheduling meals.
* **Cooking Mode** – Full-screen guided mode with timers and step navigation.
* **User Profile** – Manages preferences, saved recipes, and dietary settings.

1. **Styling**

 **CSS Framework**: TailwindCSS is used for rapid UI development and responsive design.

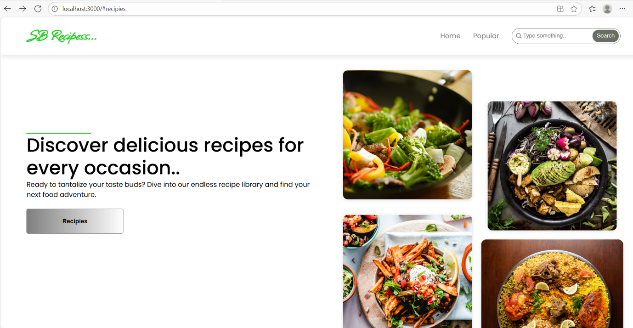
 **Theming**: Custom Tailwind configurations define colors, typography, and spacing to create a clean, modern, and food-inspired aesthetic.

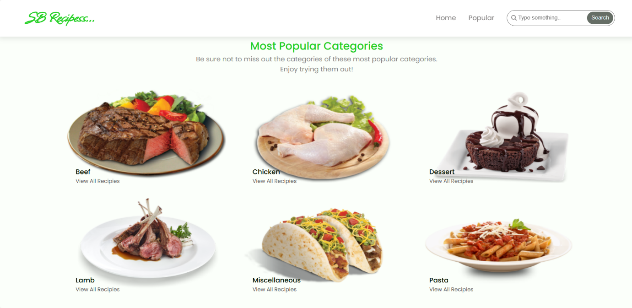
1. **Testing**

* **Testing Strategy: Unit and integration testing are implemented using Jest and React Testing Library to ensure component reliability.**
* **Code Coverage:** Default **CRA testing setup** provides baseline coverage, while additional custom test cases ensure critical features such as recipe handling, shopping list management, and cooking mode achieve high coverage levels.

1. **Screenshots or Demo**

*(Screenshots of the UI — home dashboard, recipe catalog, shopping list, meal planner, and cooking).*





1. **Known Issues**

Data persistence is limited to local storage; no cloud sync is available.

 No backend integration for multi-device access or real-time updates.

 Limited authentication and user role management.

 Recipe recommendations are basic and not fully personalized.

1. **Future Enhancements**

 Add backend API integration for cloud storage and multi-device sync.

 Implement advanced personalization with AI-based recipe recommendations.

 Introduce nutrition tracking and calorie calculators.

 Support user roles, profiles, and social recipe sharing.

 Enable voice-assisted cooking mode for hands-free guidance.

 Add export/import functionality for recipes and shopping lists.