CookBook: Your Virtual Kitchen Assistant

# 1. Introduction

POOJASRI S (TEAM LEADER)

PRABHA S

VISHNUPRIYA R

SUDHI P

# 2. Project Overview

**Purpose:**  
  
CookBook is a web application designed to assist users in exploring, organizing, and managing their culinary journey. The platform provides access to recipes, meal planning, and personalized cooking recommendations based on user preferences.  
  
**Features:**  
- Aggregation of recipes from multiple cuisines  
- Category-based filtering and keyword search  
- User customization and saved preferences  
- Interactive dashboards for meal planning  
- Responsive design for mobile and desktop

# 3. Architecture

**Component Structure**  
  
App.js – Main entry point that includes routing and layout components  
Header – Navigation bar with links to home, categories, and user profile  
RecipeFeed – Displays aggregated recipes with filtering options  
RecipeCard – Individual recipe component with metadata  
SearchBar – Allows users to search recipes by keywords  
UserPreferences – Manages user settings and personalized meal plans  
  
**State Management**  
  
The application uses Context API for global state management. User preferences, theme settings, and recipe data are shared across components using context providers.  
  
**Routing**  
Implemented using React Router v6 with routes defined for:

/ – Home page with recipe feed  
/categories – Category selection page  
/profile – User preferences and saved recipes

# 4. Setup Instructions

**Prerequisites**  
  
- Node.js version 18 or above  
- npm version 9 or above  
  
**Installation**  
Clone the repository:  
https://github.com/poojasrisenthilkumar/CookBook.git  
  
Navigate to the client directory:  
cd cookbook/client  
  
Install dependencies:  
npm install  
  
Start the development server:  
npm start

# 5. Running the Application

Run the following command in the client directory:  
  
npm start  
  
This will launch the application locally at

http://localhost:3000/.

# 6. Folder Structure

client/  
├── public/  
├── src/  
│ ├── components/  
│ │ ├── RecipeCard.js  
│ │ ├── Header.js  
│ │ ├── RecipeFeed.js  
│ │ └── SearchBar.js  
│ ├── context/  
│ │ └── AppContext.js  
│ ├── pages/  
│ │ ├── Home.js  
│ │ ├── Profile.js  
│ │ └── Categories.js  
│ ├── utils/  
│ │ └── api.js  
│ ├── App.js  
│ ├── index.js  
├── .env  
├── package.json  
└── README.md  
  
**Utilities**  
  
api.js – Handles API requests to the recipe provider  
custom hooks – useFetchRecipes for fetching recipes and useTheme for managing themes

# 7. Component Documentation

**Key Components**  
  
RecipeFeed  
- Displays recipes  
- Accepts props: filter, searchQuery  
  
RecipeCard  
- Displays a single recipe’s title, image, and summary  
- Props: recipe object  
  
UserPreferences  
- Allows users to manage their preferred cuisines and themes  
- Props: none  
  
Reusable Components  
  
SearchBar  
- Handles keyword search input  
- Props: onSearch callback  
  
Header  
- Navigation layout used across pages  
- Props: none

# 8. State Management

**Global State**  
  
- Managed using Context API  
- Holds user preferences, theme data, and API responses

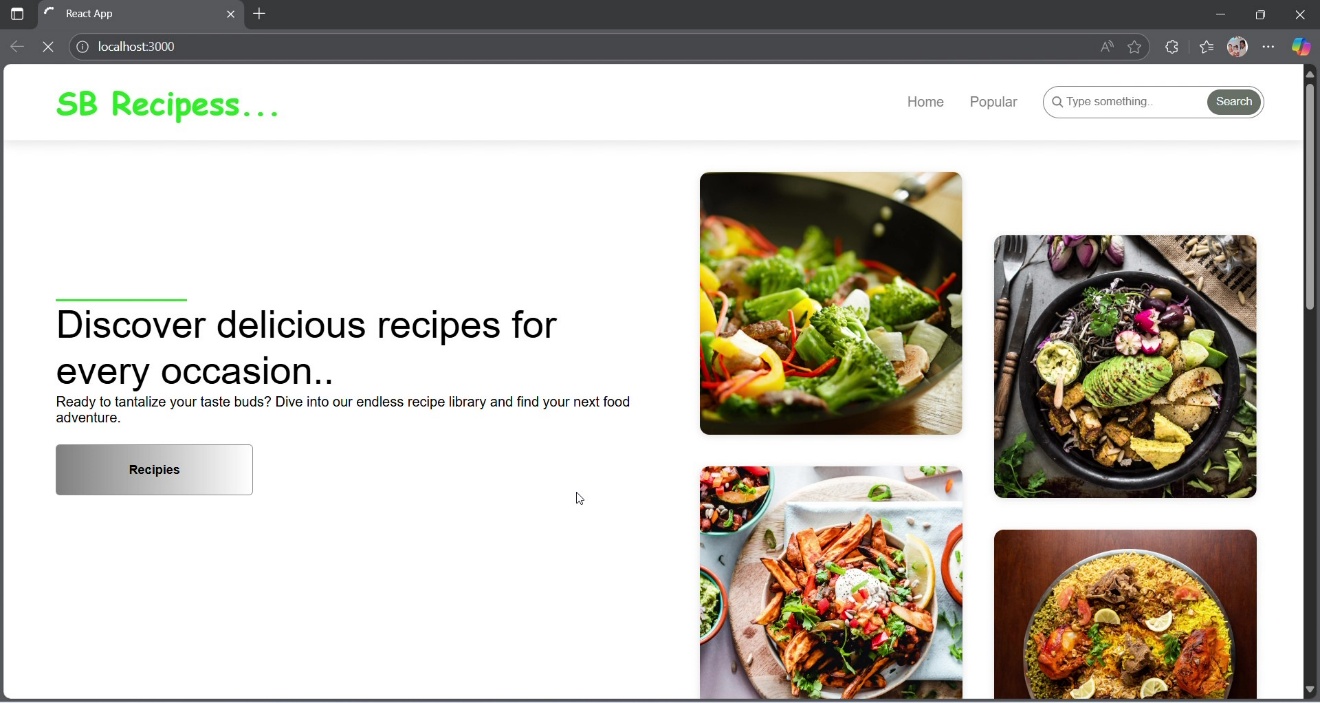
**Local State**  
- Managed using useState inside components like SearchBar for query input and RecipeCard for favorites and notes

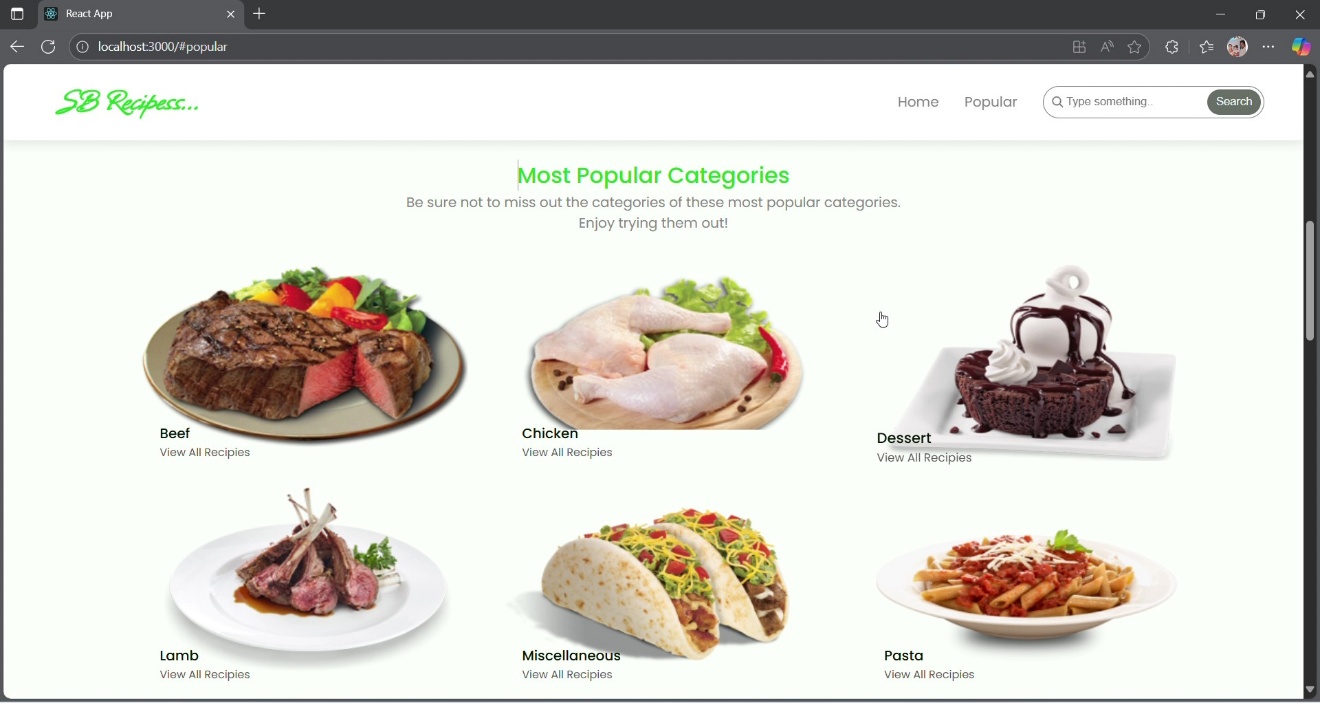
# 9. Styling

**CSS Frameworks/Libraries**  
  
- Uses Tailwind CSS for layout and styling  
- Custom styles for buttons, cards, and inputs  
  
**Theming**  
- Supports light and dark themes  
- Uses CSS variables and Context API for theme switching

# 10. User Interface

**Screenshots**  
  
- Home Page: Displays recipe feed with filters  
- Categories Page: Allows selection of preferred cuisines  
- Profile Page: User preferences and saved recipes\



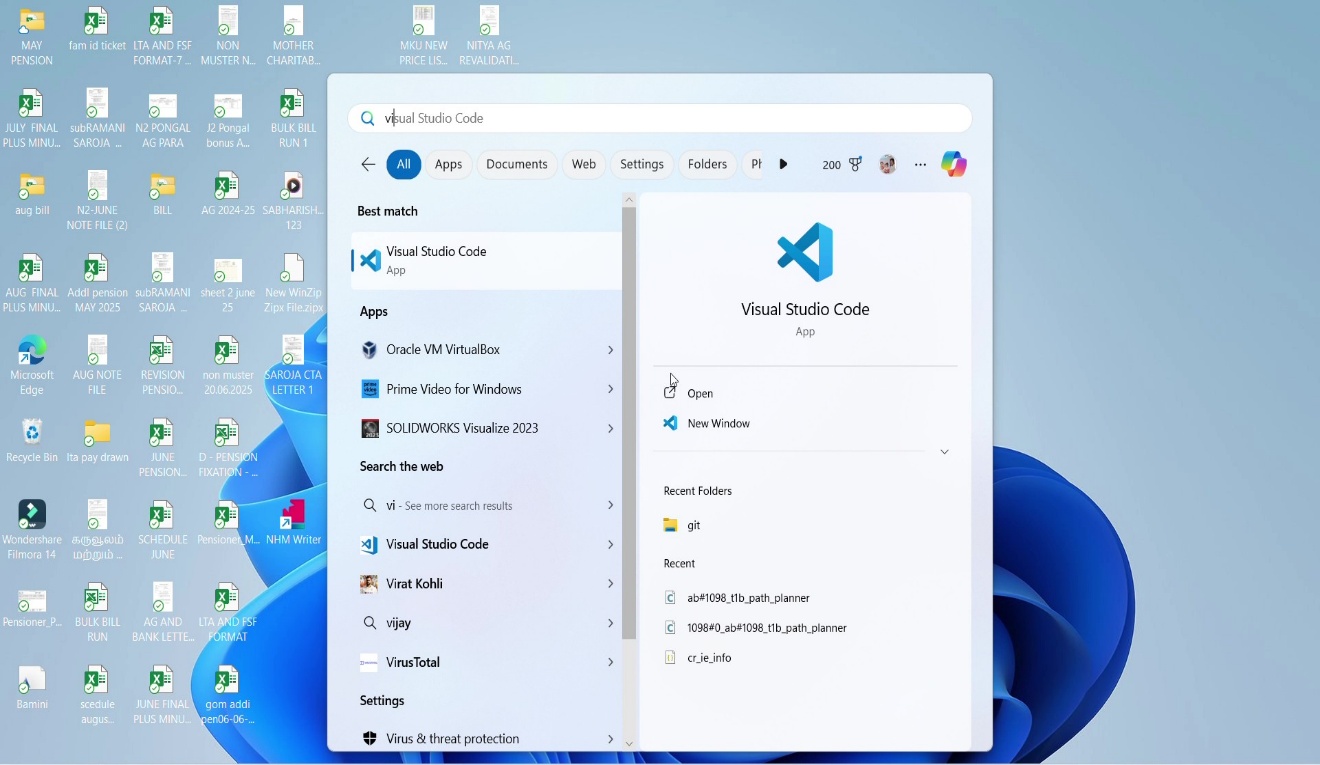


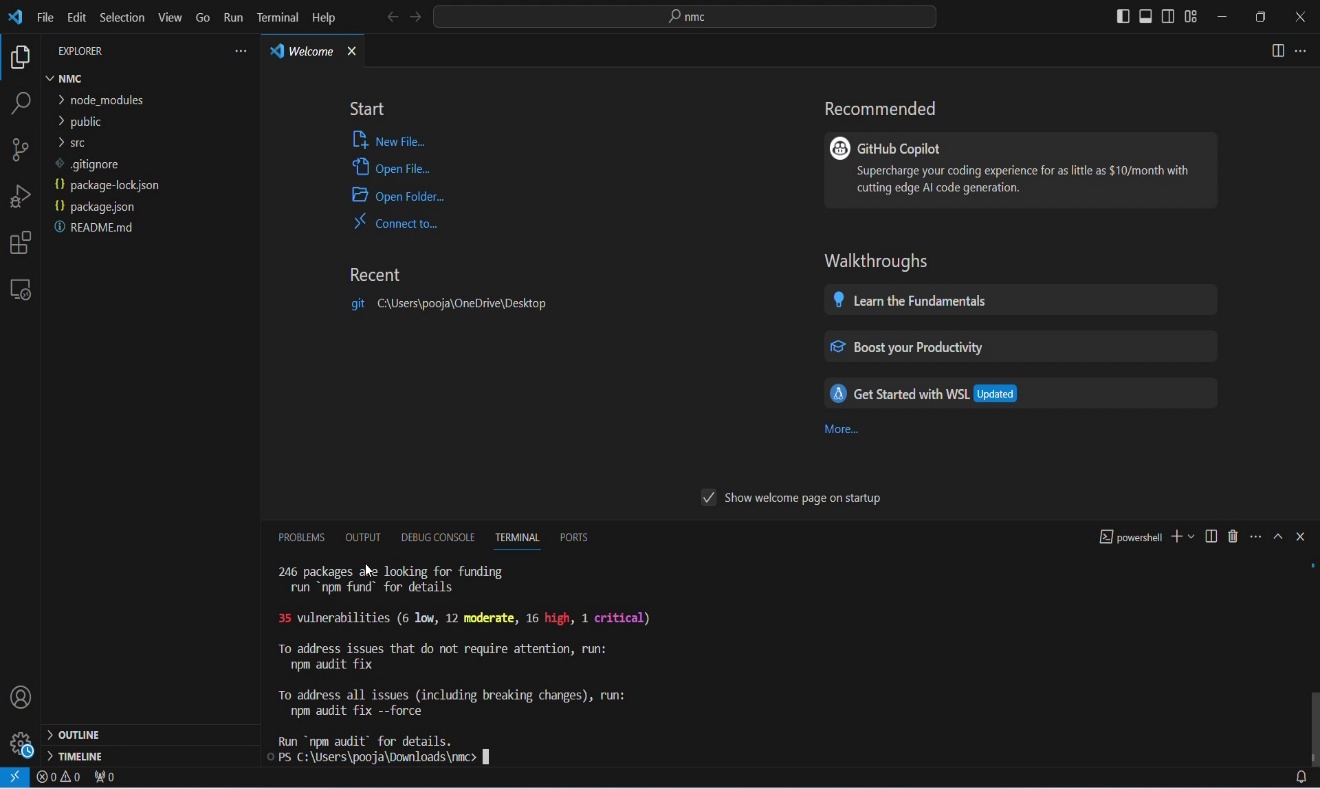
# 11. Testing

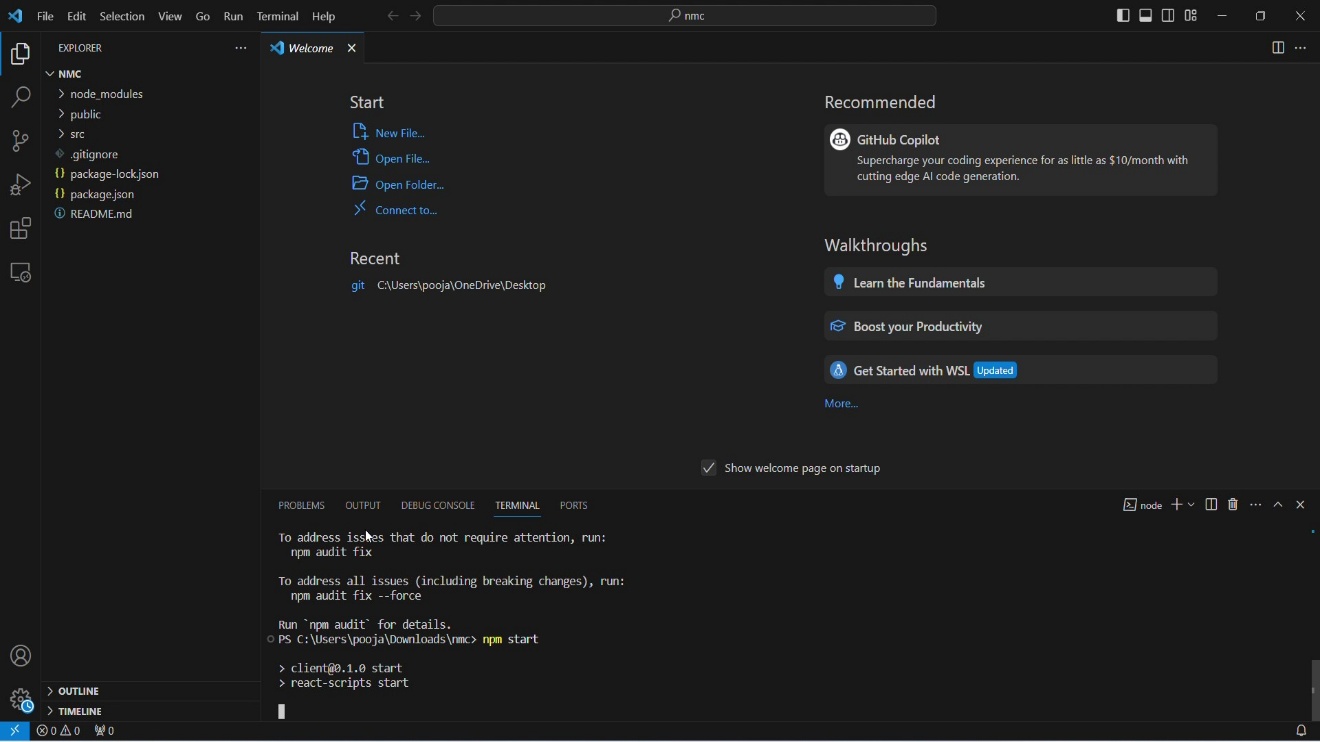
**Testing Strategy**  
  
- Unit tests with Jest  
- Integration tests using React Testing Library  
- Mock API responses for end-to-end tests  
  
**Code Coverage**  
- Ensured via Jest’s coverage reports  
- Focus on critical components like RecipeFeed and SearchBar

# 12. Screenshots or Demo

**You can view the live demo at:**  
  
<https://drive.google.com/file/d/1IIVFnr4CPbxh6rNEz9wnPv5_OE6p930f/view?usp=sharing>







# 13. Known Issues

- Occasionally slow API responses due to third-party rate limits  
- Mobile responsiveness needs minor tweaks on smaller devices  
- Some saved preferences may not persist after browser refresh

# 14. Future Enhancements

- Implement advanced filtering by cuisine, diet type, and cooking time  
- Add user authentication and recipe bookmarking  
- Introduce animations and transitions for smoother interactions  
- Optimize API calls and caching for better performance