**Cook book: Your Virtual Kitchen Assistant.**

**Introduction**

**Project Title**: **Recipe book**

**Team Members**: V MANIKANDAN [[9551590441.s@gmail.com](mailto:9551590441.s@gmail.com)] (leader)

S SATHISH KUMAR [[sathish101205@gmail.com](mailto:sathish101205@gmail.com)]

P SANJAI [[aakashp562003@gmail.com](mailto:aakashp562003@gmail.com)]

C MONISH [[monishchandru07@gmail.com](mailto:monishchandru07@gmail.com)]

**Project Overview**

**Purpose**

The **Recipe Book Project** aims to create a comprehensive and user-friendly platform for users to explore, share, and organize various recipes

**Features**

* Provide a Centralized Collection of Recipes
* Promote Cultural and Regional Dishes
* Enhance Cooking Experience Left-side menu for easy navigation
* Encourage Healthy and Budget-Friendly Cooking

**Architecture**

**Component Structure**

**Dashboard:** Displays an overview of available recipes, featured recipes, and user activity.

**Recipe List:** Shows a collection of recipes with filtering and sorting options.

**Recipe Detail:** Displays full details of a selected recipe, including ingredients, steps, andimages.

**Search & Filter:** Allows users to search recipes by nameState Management

**Routing**

* Uses **React Router** for navigation between pages

**Setup Instructions**

**Prerequisites**

* Node.js installed

**Installation**

 **Clone the repository:**

sh

CopyEdit

git clone https://github.com/SSC369/cryptoverse.git

cd cryptoverse

 **Install dependencies:**

sh

CopyEdit

npm install

 **Start the application:**

sh

CopyEdit

npm start

**Folder Structure**

* **src/components/** → Reusable React components
* **src/pages/** → Main pages of the dashboard
* **src/assets/** → Icons, images, and styles
* **src/utils/** → Helper functions and API integrations

**Running the Application**

* Run the frontend using:

sh

CopyEdit

npm start

**Component Documentation**

* **Dashboard:** Displays featured recipes, popular categories, and user activity.
* **Recipe List:** Shows a collection of recipes with filtering and sorting options.
* **Recipe Detail:** Displays full details of a selected recipe, including ingredients, instructions, and images.
* **Search & Filter:** Allows users to search for recipes by name, category, or ingredients.
* **Add Recipe:** Provides a form for users to submit their own recipes**.**
* **User Dashboard:** Displays user-specific data such as saved recipes, uploaded recipes, and user preferences**.**
* **Navigation Menu**: Provides access to different sections like Home, Recipes, Add Recipe, and Profile.
* **Authentication:** Handles user login, signup, and authentication.
* Comments & Ratings: Allows users to leave reviews and rate recipes.
* **Favorites:** Enables users to save and access their favorite recipes easily**.**

**Reusable Components**

* **Button:** Styled button for form submissions and interactions.
* **Card**: Displays recipe details (image, title, and category) in an organized manner.
* **Input Field**: Custom input fields for forms like search, login, and adding recipes.
* **Modal**: Used for pop-up confirmations (e.g., deleting a recipe or rating a recipe).
* **Loader**: Displays a loading animation when fetching data.

**State Management**

**Global State**

* Uses API calls to fetch and store recipe data from the database.
* Manages user authentication and session data.

**Local State**

* Uses React hooks (e.g., useState, useEffect) to manage component-specific states.
* Handles form inputs, filters, and search queries within individual components.

**User Interface**

* Clean and Minimalist Design: Ensures a user-friendly experience with a structured layout.
* Dark & Light Mode Support: Offers theme customization for better accessibility.
* Custom Icons & Illustrations: Uses visually appealing icons for categories and actions.
* Responsive Design: Works seamlessly on both desktop and mobile devices.

**Styling**

* Uses CSS Modules or Tailwind CSS for scoped and maintainable styles.
* Consistent Color Palette: Uses a visually appealing theme with primary, secondary, and accent colors.
* Typography & Spacing: Ensures proper readability and spacing for a clean UI.

**Testing**

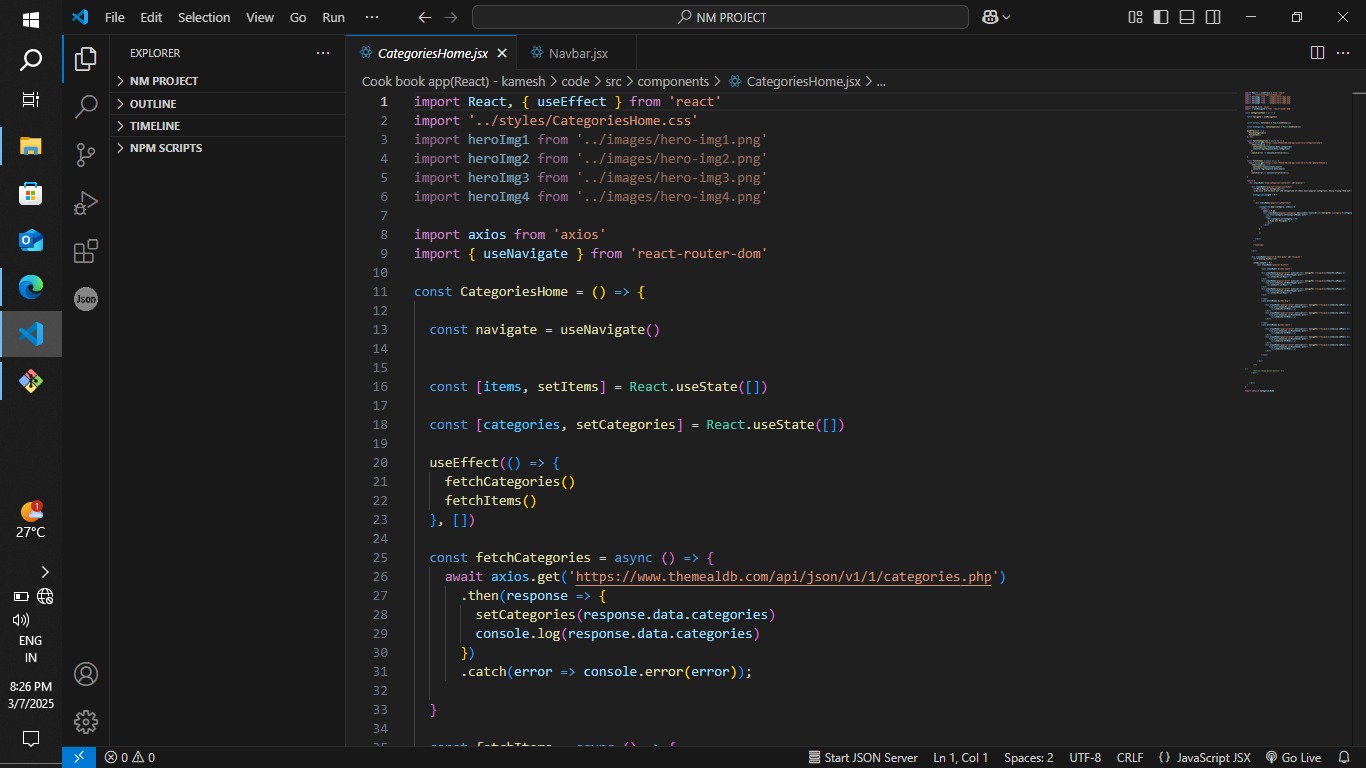
**Testing Strategy**

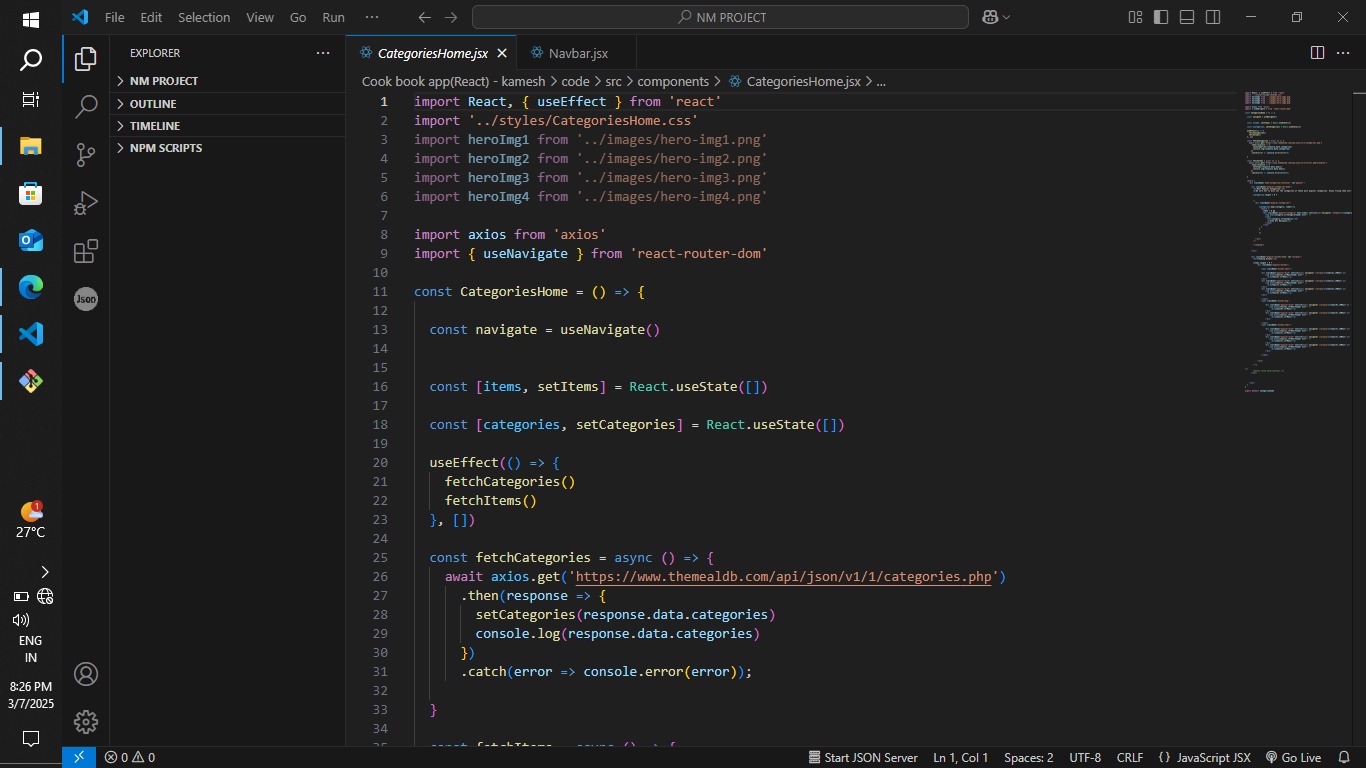
* Jest & React Testing Library for unit testing of individual components.
* Cypress or Playwright for end-to-end testing of user flows (e.g., adding a recipe, searching, logging in).
* Manual UI Testing: Ensures all elements function properly on different screen sizes.

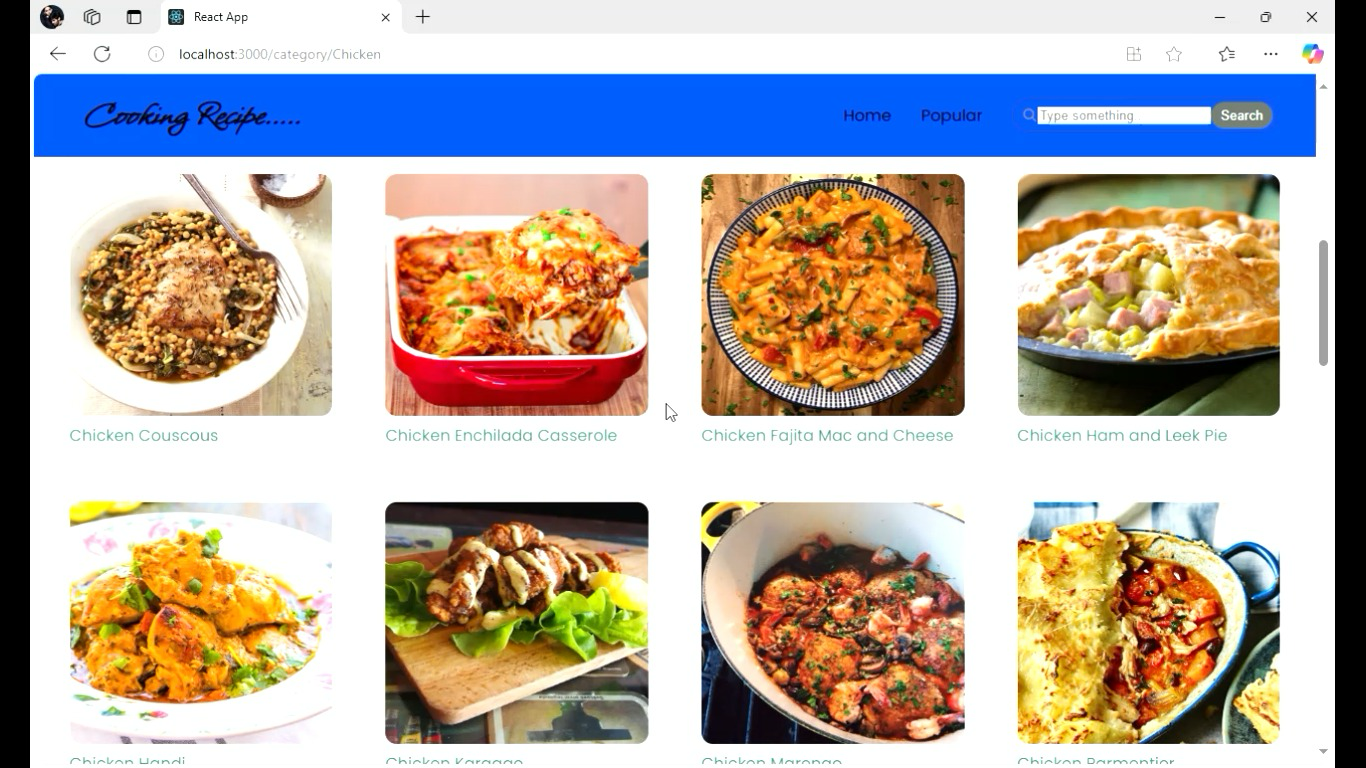
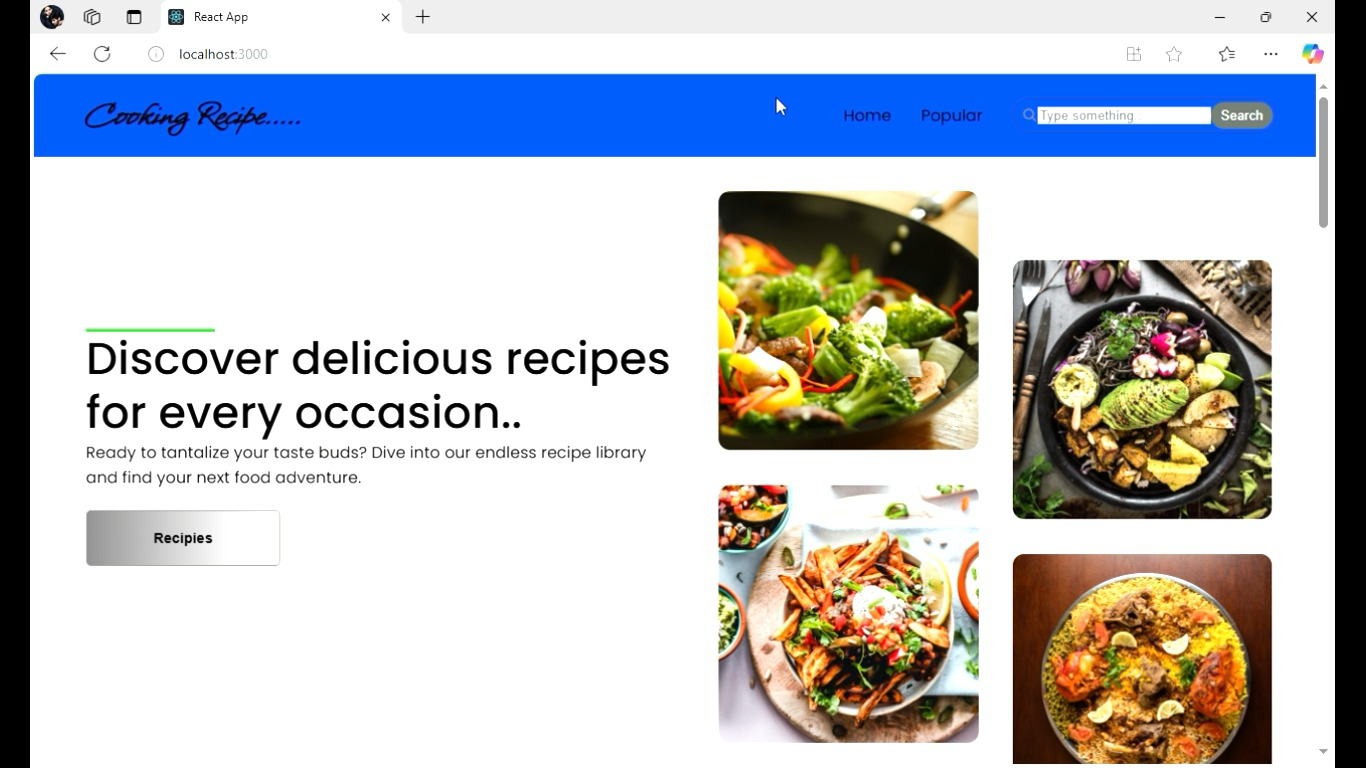
**Code Coverage**

Ensures core components are covered

**Screenshots or Demo**







**Known Issues**

* Performance optimization needed for large API responses

**Future Enhancements**

* Advanced filtering and sorting options
* More detailed analytics and reports
* User authentication for personalized settings