

PROJECT DOCUMENTATION

CookBook: Your Virtual Kitchen Assistant.

1. Introduction

- Project Title: CookBook: Your Virtual Kitchen Assistant.
- Team ID: NM22025tMID31160
- Team Leader: KRISHNAVENI K-kveni3902@gmail.com
- Team Members:
 - o JAYASRI R-jayasriravisabkar@gmail.com
 - o LAVANYA G- lavanyaganesan297@gmail.com
 - o. MOUNIKA M-mounikamuthu6383@gmail.com

2. Project Overview

- Purpose:

"CookBook is your ultimate virtual kitchen assistant, designed to make cooking easier and more enjoyable. With a vast recipe library, meal planning tools, and step-by-step cooking guidance, you'll be whipping up delicious meals in no time. Whether you're a seasoned chef or a kitchen newbie, CookBook is here to help you cook with confidence. Get cooking and make every meal a masterpiece!"

- Goals:

- i. Centralized Recipe Collection – Provide a digital platform to store and access recipes easily.
- ii. Easy Navigation – Use a clean UI and React Router for smooth browsing between categories and recipe details.
- iii. Learning Support – Integrate YouTube tutorials for step-by-step cooking guidance.
- iv. Category Organization – Group recipes by type (Beef, Chicken, Dessert, etc.) for quick discovery.
- v. User-Friendly Experience – Design an interface that is simple, responsive, and attractive.
- vi. Reusable & Scalable – Build with React components so new features and recipes can be added easily.

- Key Features:

- i. Recipe Categories – Browse recipes by popular food categories (Beef, Chicken, Dessert, etc.).
- ii. Recipe Details Page – View ingredients, preparation steps, and video tutorials.
- iii. YouTube Video Embedding – Watch cooking videos directly inside the app.
- iv. Search & Navigation – Quickly find specific recipes using smooth navigation.
- v. Responsive Design – Works on desktop, tablet, and mobile devices.
- vi. Reusable Components – Built with React components for cards, lists, forms, etc.
- vii. API Integration (if added) – Fetch recipes dynamically using Axios.
- viii. Modern UI – Styled with CSS and React Icons for a professional look.

3. Architecture

- Component Structure

- App.js – Root component, sets up routes and layout.
- Navbar.jsx – Provides site navigation across pages (Home, Categories, Recipes).
- Home.jsx – Homepage container, displays Hero, CategoriesHome, and NewsLetter.
- CategoriesHome.jsx – Shows recipe categories on the homepage.
- Category.jsx – Page that lists recipes filtered by category.

- Recipie.jsx — Page displaying a single recipe with details.
- About.jsx — Static page about the Cookbook application.
- Footer.jsx — Footer with branding, copyright, and links.
- **State Management**
 - Local State: Managed using React useState and useEffect.
 - API Integration: Axios used for fetching data from CookBook API & YouTube API.
- **Routing**
 - Library: react-router-dom
 - Routes:
 - / Home.jsx
 - /pages/Category => food Category.jsx
 - /category/Recipie ? Recipie.jsx

4. Setup Instructions

Prerequisites

- **Node.js & npm**
 - Node.js is required to run React applications.
 - npm (Node Package Manager) is used to install dependencies.
 - Download Node.js
- **React.js**
 - React is the main JavaScript library used to build this project.
 - If you don't have an existing React app, create one using:
 - npx create-react-app my-app
 - cd my-app
 - npm start
 - In SB Fitzz, the React app is already created, so you just need to install dependencies (npm install).
- **Git**
 - Used for cloning and version control.
 - Download Git
- **Code Editor**
 - Recommended: Visual Studio Code (VS Code)
 - Download VS Code
- **Basic Knowledge**
 - HTML, CSS, JavaScript
 - React concepts (components, props, hooks, state, routing)
- **Installation**
 - Get the code:
 - Download the code from the drive link given below:
 - https://drive.google.com/drive/folders/1u8PnV_mEOmwKkH_CvuNpliZtRLJZMqrO?usp=sharing
- **Install Dependencies:**
 - Navigate into the cloned repository directory and install libraries:
 - Navigate into the cloned repository directory and install libraries:
 - cd CODE
 - npm install
 - Start the Development Server:
 - To start the development server, execute the following command:
 - npm start
- **Access the App:**
 - Open your web browser and navigate to http://localhost:3000.
 - You should see the application's homepage, indicating that the installation and setup were successful.
- **Environment Variables**
 - Create a .env file with:
 - REACT_APP_API_URL=<https://exercisedb.p.rapidapi.com/exercises/equipmentList>
 - REACT_APP_YOUTUBE_API_KEY=<33cf3a7616msh4c3b1e3204f24e2p1294b3jsne16a7323d732>

5. Folder Structure

▼ public

≡ robots.txt

▼ src

> components

> images

> pages

> styles

App.css

JS App.js

JS App.test.js

index.css

JS index.js

🖼 logo.svg

JS reportWebVitals.js

JS setupTests.js

📄 .gitignore

{ } package-lock.json

{ } package.json

> node_modules

▼ public

★ favicon.ico

<> index.html

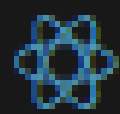
🖼 logo192.png

🖼 logo512.png

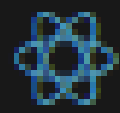
{ } manifest.json

☰ robots.txt

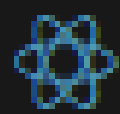
▼ components



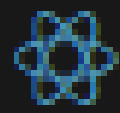
About.jsx



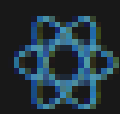
CategoriesHome.jsx



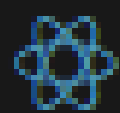
Footer.jsx



Hero.jsx

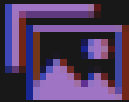


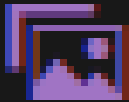
Navbar.jsx

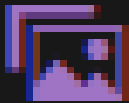


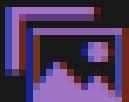
NewsLetter.jsx

✓ images

 hero-img1.png

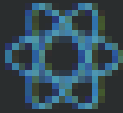
 hero-img2.png

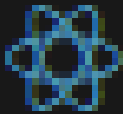
 hero-img3.png

 hero-img4.png

▼ pages

 Category.jsx

 Home.jsx

 Recipie.jsx

▼ styles

About.css

CategoriesHome.css

CategoryPage.css

Footer.css

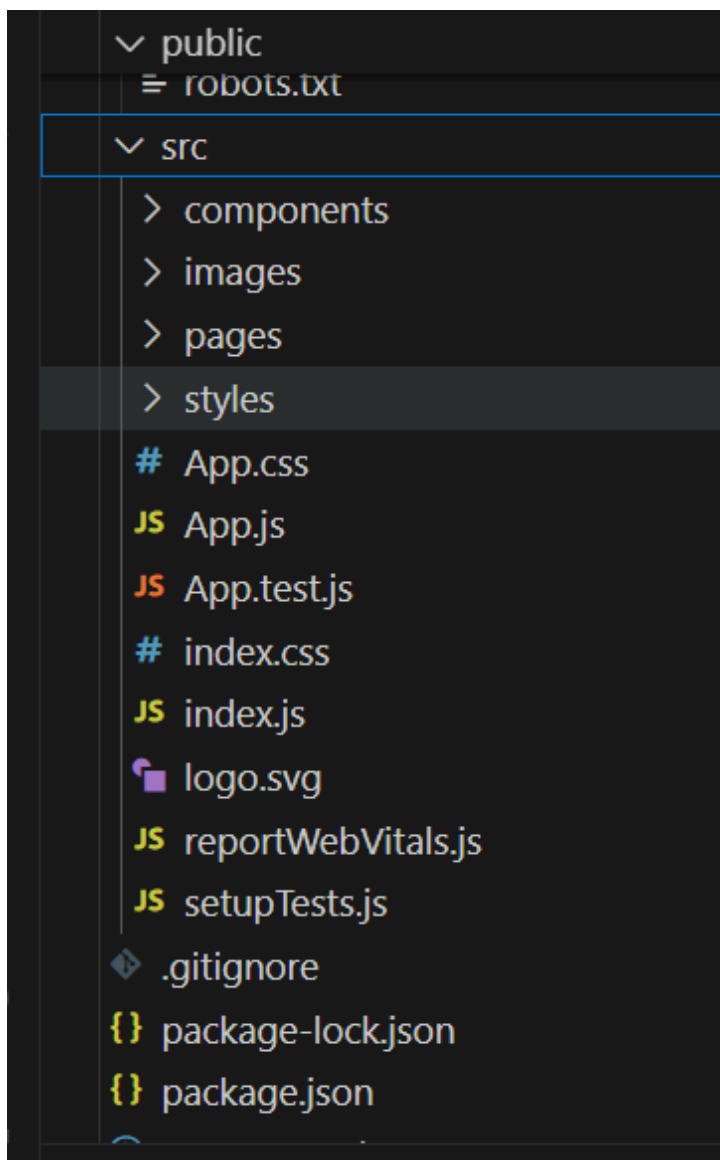
Hero.css

Home.css

Navbar.css

NewsLetter.css

Recipie.css



6. Running the Application

- Start development server:
 - npm start
- Build for production:
 - npm run build
- Run tests:
 - npm test

7. Component Documentation

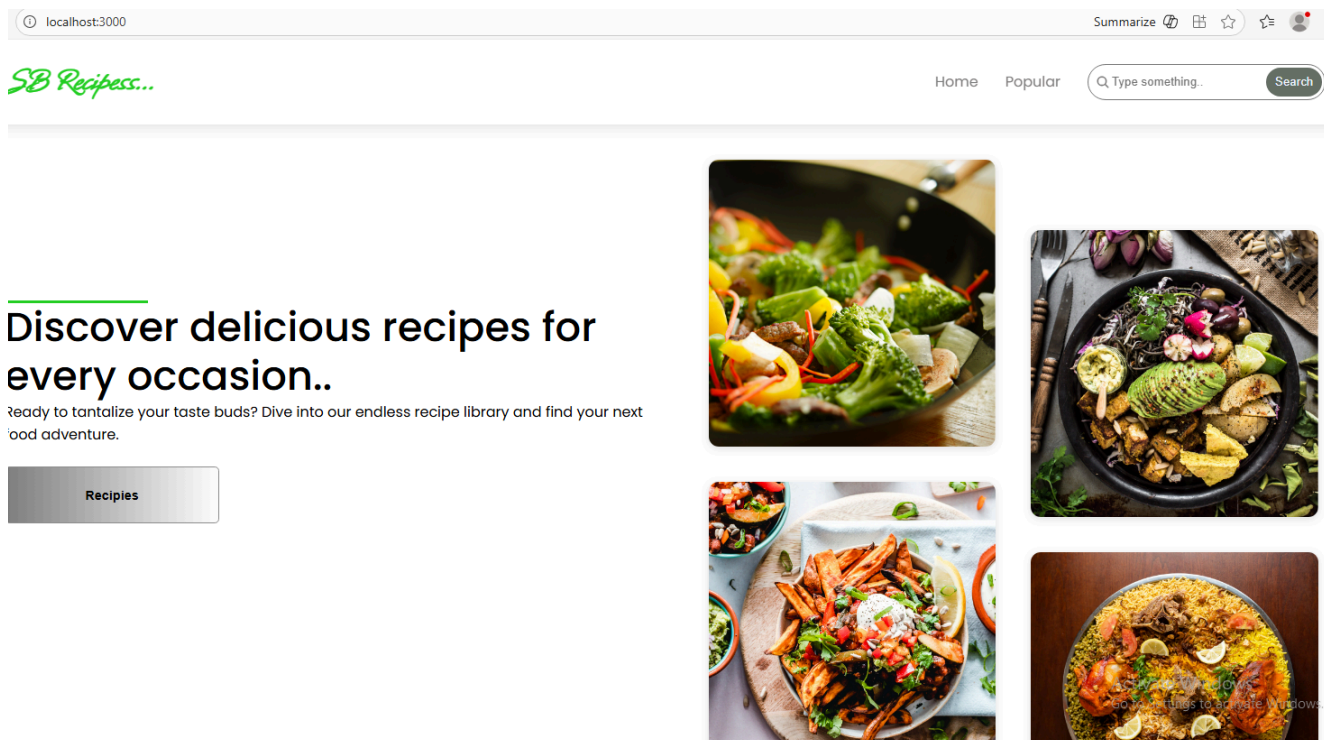
- index.js => renders <App />
- App.js => wraps Navbar, Routes, and Footer
- Home.jsx => (uses Hero, CategoriesHome, NewsLetter)
- category => Category.jsx (may also reuse CategoriesHome)
- Recipe.jsx => (shows details of a recipe) Components import their CSS modules for styling.
- images => used in Hero, Home, and Category pages

8. State Management

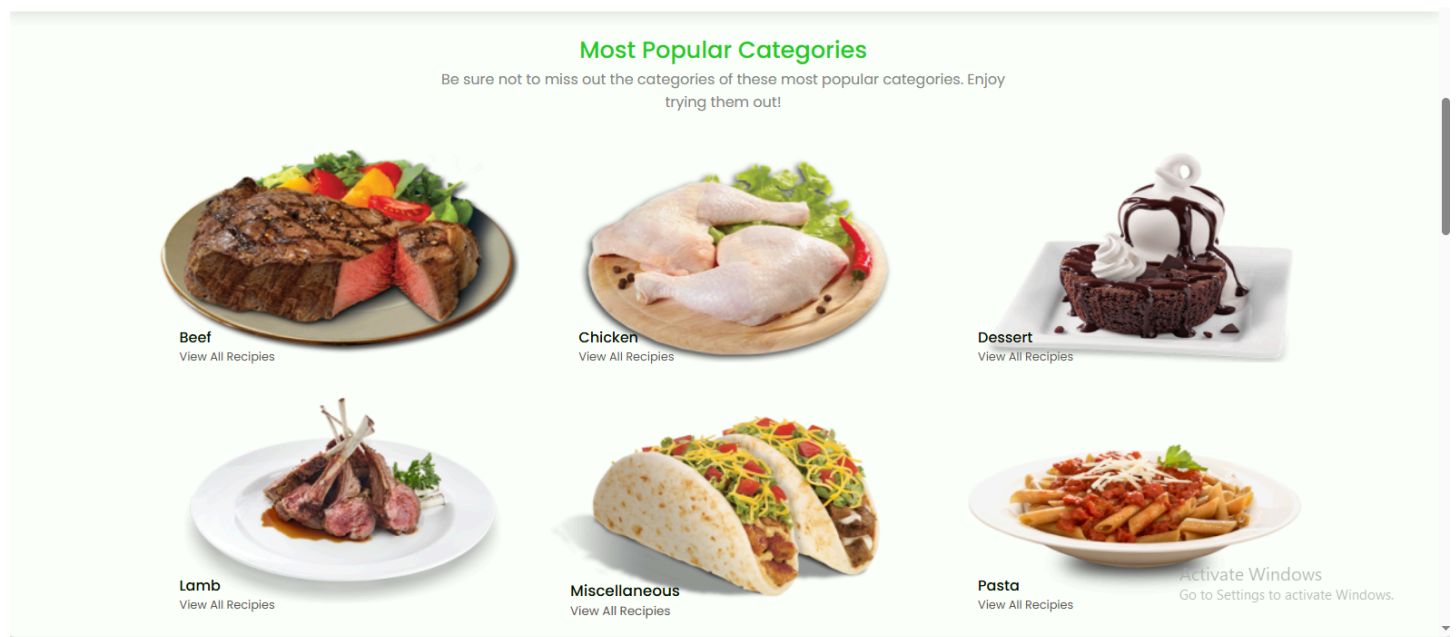
- Local State:
 - Search queries stored in HomeSearch.
 - API data fetched and stored per-page.
- Global State:
 - Not implemented — app uses component-level state.

9. User Interface

- Pages include:
 - Home (Hero + Search)



- Popular:



10. Styling

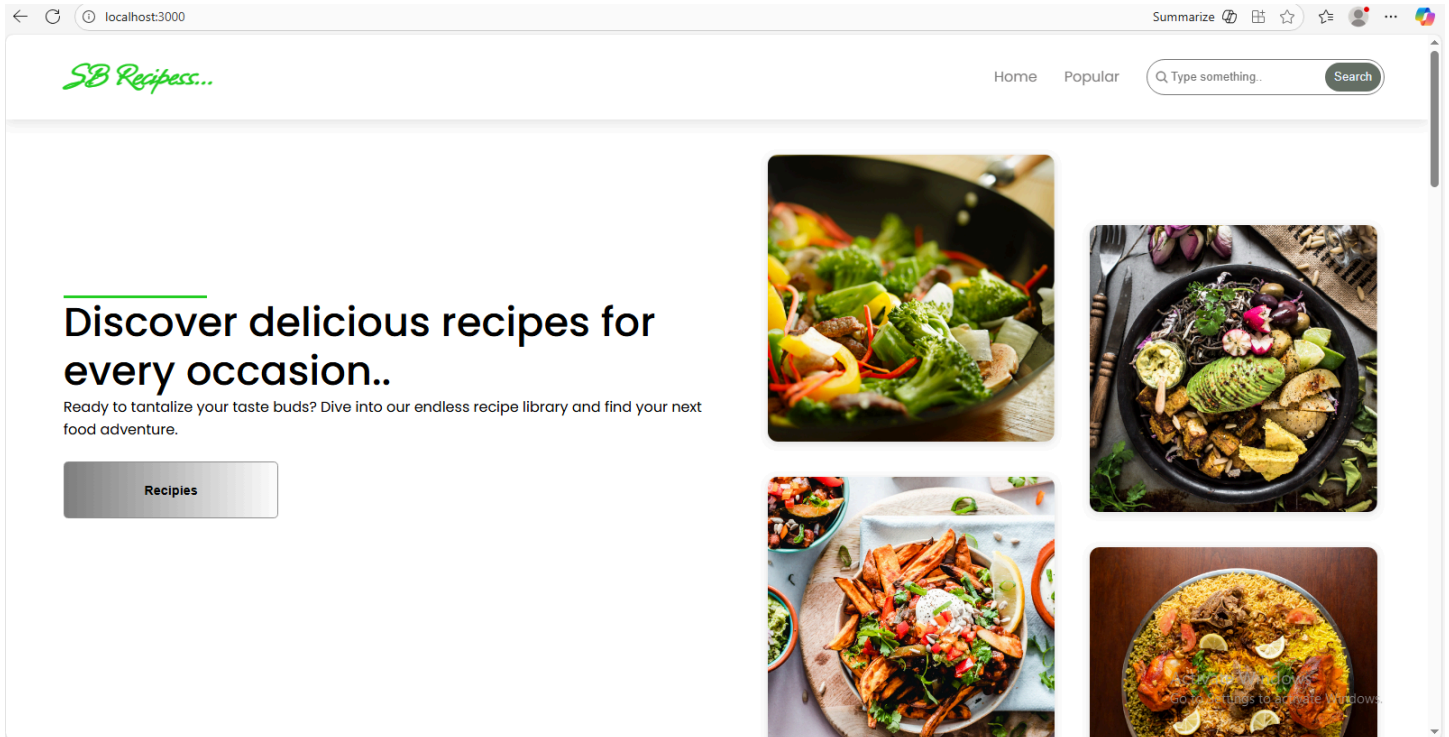
- Frameworks Used: Tailwind CSS / Bootstrap.
- Custom CSS: Stored in src/styles/.
- Each page/component has a dedicated CSS file for modularity.

11. Testing

- Libraries Used: Jest, React Testing Library.
- Unit Tests: Written in App.test.js.
- Setup: Configured with setupTests.js.

12. Screenshots / Demo

- **Demo Link:**
 - [WhatsApp Video 2025-09-08 at 5.11.46 PM.mp4](#)
- **Screenshot:**



13. Known Issues

- **API rate-limit may cause some exercises not to load.**
- **YouTube API sometimes fails to fetch related videos.**

14. Future Enhancements

- **User Authentication** – Allow users to sign up, log in, and save their favorite recipes.
- **Recipe Submission** – Let users add and share their own recipes with the community.
- **Search & Filters** – Advanced search by ingredients, cuisine, prep time, or dietary needs.
- **Favorites & Collections** – Users can bookmark recipes and organize them into collections.