***Frontend Development with React.js***

***Project Documentation for CookBook***

Your Virtual Kitchen Assistant

**(React Application)**

1. **Introduction**

* Project Title**: CookBook**

* **Team members:**
* Thamarai Selvi L**(Team leader) [Email id:** [**loguthamarai123@gmail.com**](mailto:loguthamarai123@gmail.com)**]**
* Tamilarasi T **[Email id:** [**tamilpooja47@gmail.com**](mailto:tamilpooja47@gmail.com)**]**
* Swathy R [Email id: swathyr42@gmail.com]
* Thrisha S [Email id: thrishanandhini16@gmail.com]
* Varalakshimi S [Email id: [varalakshmi11th@gmail.com](mailto:varalakshmi11th@gmail.com)]

1. **Project Overview**

* **Purpose:**

CookBook is a revolutionary web application designed to change the way you discover, organize, and create recipes. It caters to both novice and professional chefs, offering a user-friendly interface, robust features, and a vast collection of inspiring recipes.

* **Features:**
* **Recipes from the MealsDB API**: Access a vast library of international recipes spanning diverse cuisines and dietary needs.
* **Visual recipe browsing:** Explore recipe categories and discover new dishes through curated image galleries.
* **Intuitive and user-friendly design:** Navigate the app effortlessly with a clean, modern interface and clear navigation.
* **Search feature:** various dishes can be accessed easily through the search feature.

**3.Architecture**

* **Component Structure:**
* **RecipeList:**

Displays a list of all available recipes.

Each recipe links to its detailed view.

* **RecipeDetail:**

Shows detailed information about a selected recipe (ingredients, steps, etc.).

Option to save the recipe to favorites.

* **RecipeSearch:**

Allows users to search for recipes by name, ingredient, or category.

* **Favorites:**

Displays the user’s saved favorite recipes.

RecipeForm (Add/Edit Recipe):

Allows users to add new recipes or edit existing ones.

* **ShoppingList:**

Generates a shopping list based on selected ingredients from various recipes.

* **Footer:**

Contains links to other app sections (e.g., about, contact) and app info.

This structure can be expanded with additional features like user authentication or meal planning.

* **State Management:**

***Global State:*** Store the list of recipes, user preferences, and selected ingredients globally (e.g., using React's Context API or Redux).

***Local State:*** Manage form inputs and recipe filters locally in individual components using useState.

***State Persistence:*** Use localStorage or a backend to persist user data like favorite recipes and meal plans across sessions.

* **Routing:**

The application usues React Router for navigation. Routes includes:

* react-router-dom *(for routing)*
* /recipes *(route component)*
* npm install react-router-dom *(Instal)*
* react-router-dom *(to run)*

**4.Setup Instruction**

* **PRE-REQUISITES**:

* **Node.js and npm**
* **React.js**
* **HTML, CSS, and JavaScript**
* **Development Environment**
* **Get the code:**

Install Dependencies

* **Start the Development Server**:

Access the App

* **Installation:**

**Installation of required tools**:

To build CookBook, we'll need a developer's toolkit. We'll use React.js for the interactive interface, React Router Dom for seamless navigation, and Axios to fetch news data. For visual design, we'll choose either Bootstrap or Tailwind CSS for pre-built styles and icons.

Open the project folder to install necessary tools, In this project, we use:

o React Js

o React Router Dom

o React Icons

o Bootstrap/tailwind css

o Axios

For further reference, use the following resources

<https://react.dev/learn/installation>

https://react-bootstrap-v4.netlify.app/getting-started/introduction/

https://axios-http.com/docs/intro

https://reactrouter.com/en/main/start/tutorial

**5.Folder Structure:**

**Client:**

/client: React App Specific Code

This folder contains all the code related to the front-end React application.

/public:

Contains static files like index.html, icons, and images that don't need to be processed by Webpack.

/src:

This is where your React components and all other source code lives.

/components: Contains reusable, modular UI components like buttons, forms, and card layouts.

/pages: Contains different views or pages for the application (e.g., homepage, recipe detail page).

/services: Contains the logic for handling API requests or managing external data.

/context: If you’re using React Context for state management, this folder contains files related to it.

/styles: Global or component-specific styles. You can use plain CSS, SCSS, or styled-components here.

/hooks: Custom React hooks that encapsulate reusable logic, such as data fetching (useFetch) or authentication (useAuth).

/utils: Contains utility functions that help with various tasks in the app, like formatting dates or calculating nutritional values.

App.js: The root component that includes routing and overall layout for the app.

index.js: The entry point for React, where the app is rendered into the DOM.

/tests: Contains test files to validate that components and functionalities are working as expected.

**Utilities**:

Shared Logic, API Clients, and Helpers

This folder contains logic and functionality that may be used both in the client and other parts of the application. This structure is ideal for code that isn't tied to a particular React component but is essential for the overall application.

/data: This folder can hold any mock data or static resources needed for the app, like a mock list of recipes (recipesData.json).

/api: Contains shared logic for API calls. This is where you could set up an API client (e.g., using Axios or Fetch) and create functions that make calls to external services.

/helpers: Contains utility or helper functions that are more general-purpose, such as form validation or date formatting.

/config: Holds configuration files for the app, including API endpoint configurations or app-wide settings.

**6.Running the Application**

**Frontend:**

* Install Dependencies
* Start the Development Server
* Other Useful Commands
* Troubleshooting
* Optional: Connect to Back-End

**7.Component Documentation**

* **Key Component:**
* **Header** - The Header component represents the navigation bar at the top of the page.

**Props:** Reusability: Can be reused across multiple pages of the app

Improved Navigation: Provides a consistent and clear way for users to navigate and search for recipes.

* **Player** - a few things depending on the specific feature you're considering.

**Props**: To guide users through a recipe step-by-step with audio instructions. This can be useful for users who prefer audio cues while cooking.

* **Reuasibility Component**:
* **Button** - A reusable button component to handle user interactions such as submitting forms, navigating pages, and triggering actions.
* **Input**: A reusable input component to capture user data, such as text input, email, or search queries.

**Props** It ensures consistency and reusability for different input fields throughout the app.

**8.State Management**

* **Global State:**

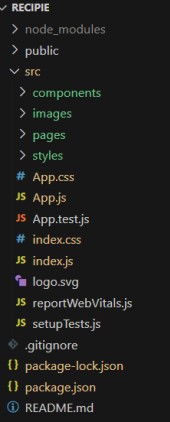
**Global state** refers to data that needs to be shared across multiple components or even the entire app. Examples of global state in a **Cookbook App** could include:

* **User Authentication**: Storing the current user's details and authentication status.
* **Recipe Data**: Keeping track of all recipes in the app and their details.
* **Theme Settings**: Global theme preferences like light/dark mode.
* **Shopping List**: Storing the list of ingredients that the user has added to their shopping list.
* **Local State:**

Local state refers to data that only needs to be available and updated within a specific component. This can include simple UI-related state like form inputs, toggle visibility, or active steps in a recipe**.**

**9 User Interface**

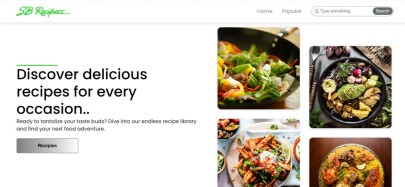
**Project structure:**



In this project, we’ve split the files into 3 major folders, *Components, Pages and Styles.* In the pages folder, we store the files that acts as pages at different url’s in the application. The components folder stores all the files, that returns the small components in the application. All the styling css files will be stored in the styles folder.

* + **Hero components**

The hero component of the application provides a brief description about our application and a button to view more recipes.



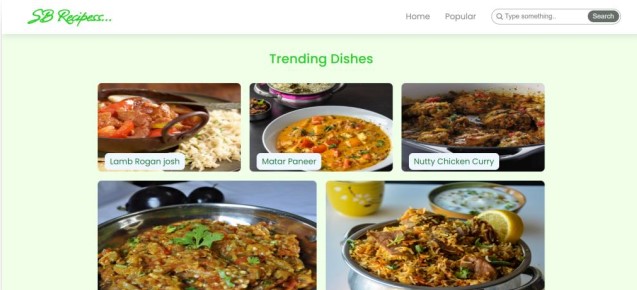
* + **Popular categories**

This component contains all the popular categories of recipes..



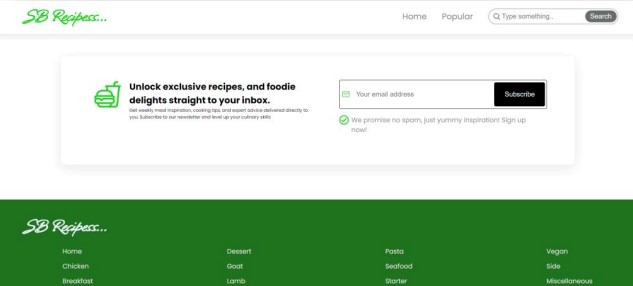
* + **Trending Dishes**

This component contains some of the trending dishes in this application.



* + **News Letter**

The news letter component provides an email input to subscribe for the recipe newsletters.



**10 .Styling**

* **CSS Frameworks/Libraries** –

This application using a CSS framework or library can significantly speed up development and provide a consistent, visually appealing design. There are several popular options to consider for your project, each with its own strengths and use cases

* **Theming –**

Theming in a Cookbook App helps ensure a consistent look and feel across your application while providing flexibility to adjust for different use cases. Whether you want to implement a light/dark mode toggle or a custom theme based on your brand's colors, a well-structured theming system will help**.**

**11.Testing**

* **Testing Strategy:**
* **Unit Testing:** Test individual units (functions, methods, components) of the app.
* **Integration Testing:** Test how different parts of the app work together (e.g., frontend-backend integration).
* **Code Coverage:**
* **Code coverage** is an important aspect of testing that helps ensure the reliability and quality of your **Cookbook App** by determining which parts of the code are being tested and which parts are not. High code coverage increases confidence in your application by identifying potentially untested areas. It’s essential to achieve a balance between code coverage and testing quality to ensure that your tests not only cover many code paths but also test meaningful cases.

**12.Screenshots or Demo**

**Project demo:**

Before starting to work on this project, let’s see the demo.

Demo link: https://drive.google.com/file/d/1khMJkccySgKyqRaEZgCpgDACHi572Llj/view?usp=sharing

Use the code in: <https://drive.google.com/drive/folders/1u8PnV_mE0mwKkH_CvuNpliZtRLJZMqrO?usp=sharing>

**Screenshots –** See section 9 for UI screenshots.

**13.Known Issues:**

**Issues 1:** Users may experience issues with the search feature where the search results either do not return relevant recipes or fail to return any recipes at all.

**Issues 2:** Another issue users might face is that recipe images either don’t load at all or are displayed incorrectly (e.g., broken links, incorrect image formats, or images that don’t scale properly on different devices).

**14.Future Enhancements**

* **Future Features:**

1. Meal Planner and Grocery List Integration
2. User-Generated Recipe Reviews and Ratings
3. Cooking Mode with Timer and Step-by-Step Guidance
4. Nutritional Information and Dietary Filters
5. Cooking Video Tutorials
6. Smart Recipe Suggestions Based on Available Ingredients
7. Social Sharing and Cooking Challenges
8. Personalized Recipe Recommendations

This documentation provides a comprehensive overview of the CookBook project, including its architecture, setup instruction,and future plans.