**Frontend Development with React.js**

**1. Introduction**

* **Project Title**: COOK BOOK (Recipe App)
* **Team Members**:
  + **Team Leader**: PRIYA P
  + **Team Member**: NAGAIYARASI V
  + **Team Member**: NANDHINI D
  + **Team Member**: PARVATHI G
  + **Team Member**: SANDHIYA R

**Description**:  
The Recipe App is designed as a comprehensive platform for food enthusiasts, home cooks, and anyone looking to explore and share culinary delights. The app allows users to discover, save, and share recipes, as well as interact with the community through ratings, reviews, and comments.

**2. Project Overview**

**Purpose:**

The main purpose of the Recipe App is to create a dynamic and interactive platform where users can search, explore, and share recipes easily. It aims to connect home cooks, foodies, and professional chefs to discover culinary ideas, rate recipes, and share their own creations with a larger audience.

**Features:**

* **Recipe Browsing**: Users can browse recipes by categories, cuisine types, or ingredients.
* **Recipe Submission**: Users can submit their own recipes with detailed instructions, ingredients, and images.
* **Recipe Search**: Search functionality to find recipes based on keywords.
* **User Profiles**: Users can create and manage their profiles, view their favorite recipes, and saved recipes.
* **Recipe Ratings & Reviews**: Users can rate recipes and leave comments on them.
* **Responsive UI**: The application is optimized for both desktop and mobile devices.

**3. Architecture**

**Component Structure:**

The application is designed using a modular component-based architecture:

* **HomePage**: Displays featured recipes and categories.
* **RecipeCard**: Represents each recipe summary (title, image, rating).
* **RecipeDetail**: Displays detailed recipe instructions, ingredients, and user comments.
* **RecipeForm**: Form to add new recipes.
* **SearchBar**: A component to search recipes by name or ingredient.
* **UserProfile**: Displays the user's saved recipes and details.

**State Management:**

The app uses **React Context API** for global state management to share data like user details and favorite recipes across components.

* **Global State**:
  + User authentication status.
  + Favorite recipes.
  + Recipe collection.
* **Local State**:
  + Form data for adding new recipes.
  + Search query for filtering recipes.

**Routing:**

**React Router** is used for navigation:

* / - Home page showing featured recipes and categories.
* /recipe/:id - Recipe detail page.
* /add-recipe - Page for submitting a new recipe.
* /profile - User profile page showing saved recipes and details.

**4. Setup Instructions**

**Prerequisites:**

* **Node.js** (v14 or above)
* **npm** (Node Package Manager)

**Installation:**

1. Clone the repository:
2. git clone https://github.com/username/cookbook-recipe-app.git
3. Navigate to the project directory:
4. cd cookbook-recipe-app
5. Install dependencies:
6. npm install
7. Set up environment variables in .env file:
8. REACT\_APP\_API\_URL=http://your-api-url.com

**5. Folder Structure**

**Client:**

* **/src**
  + **/components**: Reusable components like RecipeCard, Button, SearchBar.
  + **/pages**: Main page components like HomePage, RecipeDetail, AddRecipe.
  + **/assets**: Static files like images, fonts.
  + **/utils**: Utility functions and hooks like useFetch, useAuth.

**Utilities:**

* **useFetch**: Custom hook to fetch recipes from the API.
* **useAuth**: Custom hook to manage user authentication and profile state.

**6. Running the Application**

To run the app locally:

1. Start the development server:
2. npm start
3. Visit http://localhost:3000 in your browser.

**7. Component Documentation**

**Key Components:**

* **RecipeCard**:
  + **Props**:
    - title: The name of the recipe.
    - image: Image URL for the recipe.
    - rating: Rating of the recipe.
    - onClick: Event handler to navigate to the recipe details page.
  + **Purpose**: Displays a card for each recipe with a title, image, and rating.
* **RecipeForm**:
  + **Props**:
    - onSubmit: Callback function to handle form submission.
  + **Purpose**: A form to add new recipes, including inputs for ingredients, instructions, and an image upload.

**Reusable Components:**

* **Button**:
  + **Props**:
    - onClick: Callback for button click.
    - label: The text displayed on the button.
    - type: Button type (e.g., primary, secondary).
  + **Purpose**: Reusable button component used across the app.
* **SearchBar**:
  + **Props**:
    - query: Current search query.
    - onChange: Event handler for updating the search query.
  + **Purpose**: Search bar for filtering recipes.

**8. State Management**

**Global State:**

* **User Context**: Handles user authentication state (logged in or not) and profile information.
* **Recipe Context**: Manages recipes, favorites, and recent search history across different components.

**Local State:**

* Each component like RecipeForm and SearchBar uses **React’s useState** to handle local state, such as form inputs or search queries.

**9. User Interface**

**UI Features:**

* **Recipe Search**: A search bar at the top allows users to filter recipes by ingredients or name.
* **Recipe Cards**: Each recipe is shown in a card layout with the title, image, and rating.
* **Recipe Detail Page**: Shows detailed recipe information, including ingredients and user comments.
* **User Profile**: Displays the list of recipes saved by the user along with their profile picture.

**Screenshots:**

(Include relevant screenshots showing UI elements like homepage, recipe cards, and profile page.)

**10. Styling**

**CSS Frameworks/Libraries:**

* **Styled-Components**: Used for scoped and dynamic styling of components.
* **CSS Modules**: For component-specific styles that are modular and prevent global styling issues.

**Theming:**

* **Light and Dark Themes**: The app supports light and dark modes, which users can switch between. Custom design tokens are used for consistent color, typography, and layout across the app.

**11. Testing**

**Testing Strategy:**

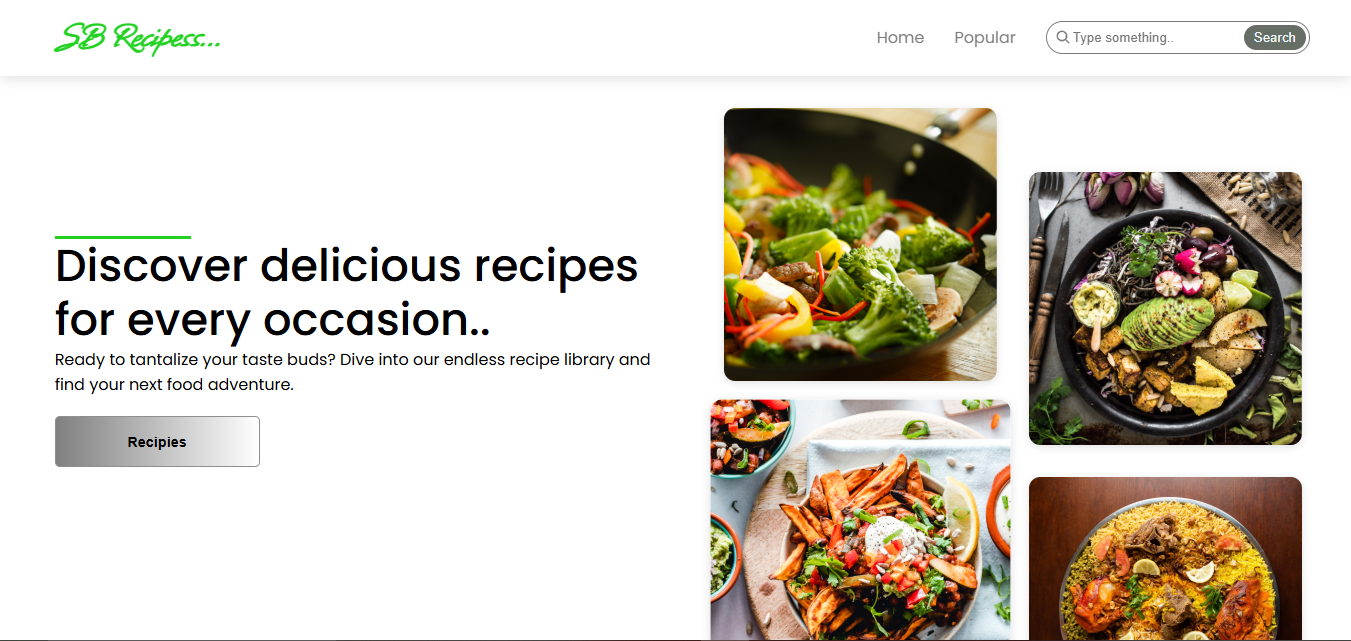
* **Unit Testing**: Using **Jest** and **React Testing Library** to test individual components like buttons, forms, and lists.
* **Integration Testing**: Ensuring that components work together seamlessly (e.g., submitting a form updates the global state).
* **End-to-End Testing**: Using **Cypress** for testing the entire user journey, from logging in to adding a recipe and searching for it.

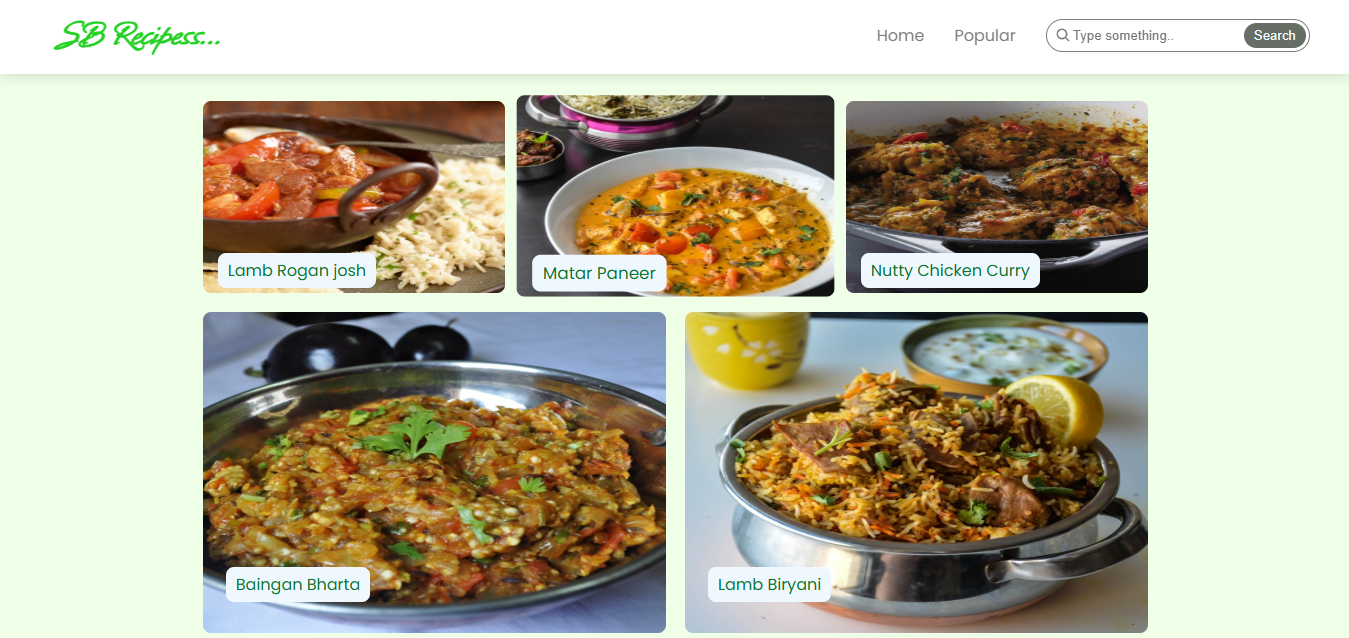
**Code Coverage:**

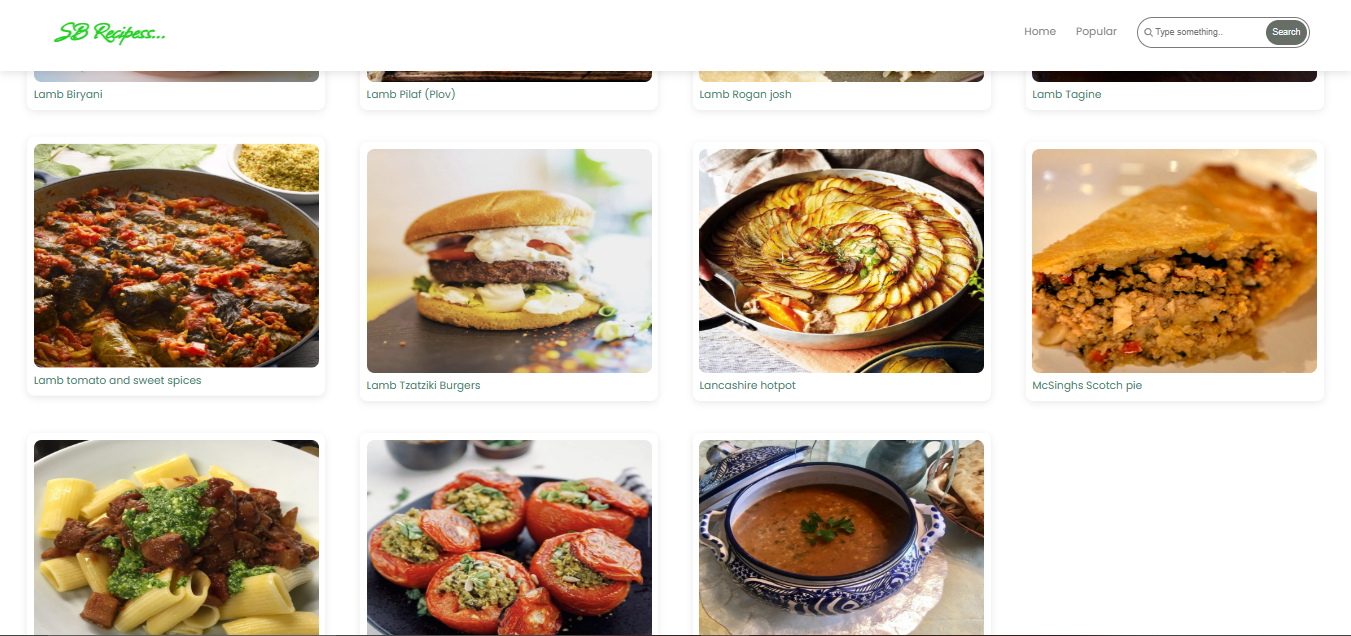
* **Jest Coverage**: Code coverage is enforced using Jest’s built-in coverage tool to ensure all critical code paths are tested.

**12. Screenshots or Demo**

(Provide demo links or images of the app showcasing key features and UI elements.)







**13. Known Issues**

* **Search Filtering**: Sometimes, the search feature may return results even when no recipes match the query.
* **Recipe Submission**: Users occasionally face issues with uploading large images for new recipes.

**14. Future Enhancements**

* **Advanced Filters**: Adding additional filters for searching, such as cooking time, difficulty, and ingredient count.
* **Recipe Suggestions**: Implementing a personalized recommendation system based on user preferences.
* **Notifications**: Push notifications for new recipe submissions or comments on user’s recipes.
* **Mobile App**: A dedicated mobile version of the app for Android and iOS using React Native.

This document provides an in-depth overview of the **Recipe App**'s architecture, setup, and design, making it easy for the team to understand the app's structure and development process.