**Frontend Development with React.js**

*Project Documentation format*

1. **Introduction:**

         o **Project Title :** *COOK BOOK*

         o **Team Members and Roles:**

| **Name** | **Role** |
| --- | --- |
| *H.Haniya Tasneem* | *Frontend developer* |
| *J.Mufeedha* | *Content manager* |
| *M.Rasmiya* | *Frontend designer* |
| *M.Mehanas* | *Document creator* |

**2. Project Overview:**

o **Purpose:**

*The goal of this Cookbook application is to provide a modern, digital platform that brings together cooking enthusiasts, home chefs, and food lovers. The app serves as a one-stop solution for discovering, organizing, and sharing recipes across a wide range of cuisines and such preferences.*

o Features:

**🧭 *Homepage***

* *Highlights featured recipes, trending dishes, and seasonal picks.*
* *Clean and welcoming layout with hero banner and calls to action.*

**📄Recipe Browsing**

*Users can view a wide selection of recipes categorized by cuisine, meal type, difficulty level, and more.*

**🔍Search and Filter:**

*A dynamic search bar and filters help users quickly find specific recipes based on ingredients, cooking time, or dietary preferences*.

**🍽️Recipe Details Page:**

*Displays ingredients, step-by-step instructions, preparation time, servings, and images for each recipe.*

**📱 Responsive Design**  
              *Fully optimized for desktops, tablets, and mobile devices for a smooth user experience across all platforms.*

**🍲 Easy access** to a wide range of recipes.

**3. Architecture :**

***Component Structure:***

*The application is built with modular React components for clarity and reusability. Key components include:*

* ***web:****The root component managing the overall layout.*
* ***Navbar:*** *Provides navigation across different sections.*
* ***RecipeList:*** *Displays a list of recipes filtered or searched by the user.*
* ***RecipeCard:****Shows a brief overview of each recipe in the list.*
* ***RecipeDetails:****Shows detailed information about a selected recipe, including ingredients and cooking steps.*

*These components communicate through props to pass data and handle user interactions.*

**State Management:**

*State is managed locally within components for UI elements such as search. Global state management is minimal and handled with React’s built-in state hooks*

**Routing:**

*The app uses****React Router****to handle navigation, with routes such as:*

* */ — Home page showing the recipe list.*
* */recipe— Details page for a selected recipe.*

**4. Setup Instructions :**

**Prerequisites:**

*Before setting up the project, ensure you have the following installed on your system:*

  ***Node.js****— for running JavaScript on the backend and managing packages*

  ***npm****(comes with Node.js) or****yarn****— package managers to install dependencies*

  *A code editor like VS Code.*

**Installation*:***

 ***Clone the Repository:*** *Open your terminal and run*

 *git clone*[*https://github.com/your-username/your-cookbook-repo.git*](https://github.com/your-username/your-cookbook-repo.git)

*Replace the URL with your actual repository link.*

 *Navigate to the Project Directory*

 *cd your-cookbook-repo*

 ***Install Dependencies:*** *Using npm:*

*npm install*

*Or using yarn:*

*yarn install*

 ***Configure Environment Variables (if any):*** *If your project uses environment variables, create a .env file in the root directory and add your configurations. For example:*

 *REACT\_APP\_API\_URL=*[*https://api.yourcookbook.com*](https://api.yourcookbook.com/)

 ***Start the Development Server:***

 *npm start*

**5. Folder Structure :**

**Client:**

*The React application is organized into the following main folders for better modularity and maintainability:*

* ***/src/components*** *Contains reusable UI components such as Navbar, RecipeCard, RecipeList, and RecipeDetails.*
* ***/src/pages*** *Includes page-level components representing different like HomePage and RecipePage.*
* ***/src****/hero  
  Stores static files like images, icons, and stylesheets.*
* ***/src/styles*** *Contains CSS or styling files, including global styles and theme-related files.*
* ***/src/utils*** *Includes utility functions that perform common operations like formatting , filtering.*
* ***/src/App.js*** *The root React component that configures routing and app layout.*
* ***/src/index.js*** *The entry point that renders the React app.*

**Utilities:**

*The project includes several helper functions and custom hooks to keep the code clean and maintainable:*

* ***Helper Functions:*** *Utility functions in /src/utils handle tasks such as:*
  + *cooking video tutorial*
  + *recipes based on non veg and veg*
* ***Custom :*** *Hooks in /src/ encapsulate reusable logic, such as:*
  + *useRecipe : recipe data from an API or local JSON file.*
  + *useSearchFilter: Manages the state and logic for searching and  recipes.*

**6. Running the Application :**

**Frontend:**

*To start the React frontend server locally, follow these steps:*

 *Open your terminal and navigate to the project’s client directory:*

 *cd code*

 *Run the development server with:*

 *npm start*

**7. Component Documentation**

**Key Components:**

* ***web***

o    ***Purpose:****The root component that sets up routing and general layout for the application.*

* ***Navbar***

o    ***Purpose:****Displays the navigation bar with links to different pages and the theme toggle.*

 ***RecipeList***

o    ***Purpose:****Shows a list of recipes based on search or filters applied by the user.*

  *recipes : An array of recipe objects to display.*

  *onSelectRecipe : Callback triggered when a user clicks on a recipe to view details.*

 ***RecipeCard***

o    ***Purpose:****Displays a summary of a single recipe including image, title, and brief info.*

  *recipe : Recipe data object containing details like title, image URL, and description.*

 ***RecipeDetails***

o    ***Purpose:****Shows detailed information for a selected recipe, including ingredients and step-by-step instructions.*

  *recipe : Full recipe data to display.*

**Components:**

* **Button**

*\*A button labeled****About****is displayed.*

*\* The function shows a popup with a message.*

* **purpose**
  + *A customizable button used across the app for actions like navigation.*
    - *text displayed on the button.*
    - *click event handler.*

***SearchBar***

* + ***Purpose:****Input field for searching recipes by keywords.*

***6.*Running the Frontend Server Locally :**

        **Open your terminal.**

        **Navigate to the frontend directory**, usually named client or something similar:

cd client

        **Start the frontend server** with the following command:

        npm start

This will run the frontend React app (or whichever frontend framework you’re using) locally, usually accessible at [http://localhost:3000](http://localhost:3000/).

**7. Component Documentation :**

**Key Components :**

 ***RecipeList***

* ***Purpose:****Displays a list of recipes fetched from the backend or state.*
* ***Props:***
  + *Recipes\_ An array of recipe objects to display.*
  + *onSelectRecipe— Callback function triggered when a recipe is selected.*

 ***RecipeDetail***

* ***Purpose:****Shows detailed information about a selected recipe, including ingredients, steps,videos and images.*
* ***Props:***
  + *recipe— The recipe object containing all details.*
  + *onBack \_ Callback function to navigate back to the recipe list.*

 ***AddRecipeForm***

* ***Purpose:****Form component for adding a new recipe to the cookbook.*
* ***Props:***
  + *onSubmit — Callback triggered when the form is submitted.*
  + *initialData  — Pre-filled data for editing existing recipes.*

***Components :***

 ***Button***

* ***Purpose:****A styled* button component used across the app.
* **Props:**
  + *Text to display inside the button.*
  + *Event handler for click events.*
  + *type— Button type attribute ("button", "submit", etc.). Default: "button".*
  + *disabled — Whether the button is disabled.*

 **InputField**

* **Purpose:** A reusable input field component for text inputs.
* **Props:**
  + *type (String, optional) — Input type ("search”,”email")*

 **Modal**

* **Purpose:** *A reusable modal dialog for confirmation messages, forms, or alerts.*
* ***Props:***
  + *title (String) — Modal title text.*

**8. State Management :**

**Global State:**

*The application uses React’s built-in****Context API****to manage global states that need to be accessible across multiple components. For example:*

* ***Theme Preference:****Tracks whether the user has selected light or dark mode, allowing the entire app to update styles accordingly.*
* ***Search and Filter Criteria:****Stored globally so that multiple components (like Navbar, RecipeList,) can access and update the current search term and filter selections.*

*The global state is provided via a Context Provider at a high level allowing child components to consume and update shared state using like some menus*

**Local State:**

*Components manage their own internal states using React’s useState  for UI-specific data. Examples include:*

* ***RecipeList:****Manages pagination or temporary loading states.*
* ***SearchBar:****Holds the current input value before updating the global search state.*
* ***RecipeDetails:****Tracks UI behavior such as expanding or collapsing instruction steps.*

*This separation ensures that only necessary state is global, keeping the app performant and easier to maintain.*

**9. User Interface :**

**->Home Page / Recipe List**

*Description: The home page displays a list of all recipes available in the cookbook. Users can scroll through recipes and click on any recipe to view details.*

***->Recipe Detail Page***

*Description: This page shows the selected recipe's ingredients, preparation steps, images. It allows users to go back to the recipe list or edit the recipe.*

***->How to Capture Screenshots and GIFs:***

* *Use tools like****Snipping Tool****screenshots.*
* *Upload the files to your project or documentation folder and link them as shown.*

***10. Styling***

***>>*CSS Frameworks/Libraries<<**

* ***Styled-Components:*** *The project uses****Styled-Components****, a CSS-in-JS library that allows styling React components with scoped, dynamic styles directly inside JavaScript files. This approach promotes component-level styling and easier theming.*
* ***Normalize.css:*** *To ensure consistent styling across different browsers,****Normalize.css****is included to reset default browser styles.*
* ***Optional: Sass (if used):*** *If the project uses****Sass****as a preprocessor, mention it here — it helps organize styles with variables, nesting, and mixins, improving maintainability.*

***>>Theming<<***

* ***Custom Theme System:*** *The application implements a custom theme using****Styled-Components’ ThemeProvider****. This allows defining global color palettes, font styles, and spacing scales that components consume for consistent stylish.*

***11. Testing***

***Testing Strategy***

* ***Unit Testing:*** *Individual components and utility functions are tested using Tests focus on rendering correctness, function outputs, and edge cases to ensure components behave as expected in isolation.*
* ***Component Testing:*** *The project leverages****React Testing Library****for testing React components. This library encourages testing from the user’s perspective by interacting with UI elements, ensuring that components render correctly and handle user interactions properly.*
* ***Integration Testing:*** *Some tests cover integration scenarios where multiple components interact, such as form submission flows or data fetching in the recipe list.*
* *simulate full user journeys through the application. This verifies that the main workflows (e.g., adding a recipe, editing, and deleting) function* *Reports on test coverage, highlighting the percentage of lines, functions,*
* *Coverage reports are regularly reviewed to ensure critical components and logic branches are adequately tested.*
* *Thresholds can be configured to prevent merges if test coverage drops below of such.*

***Example Commands***

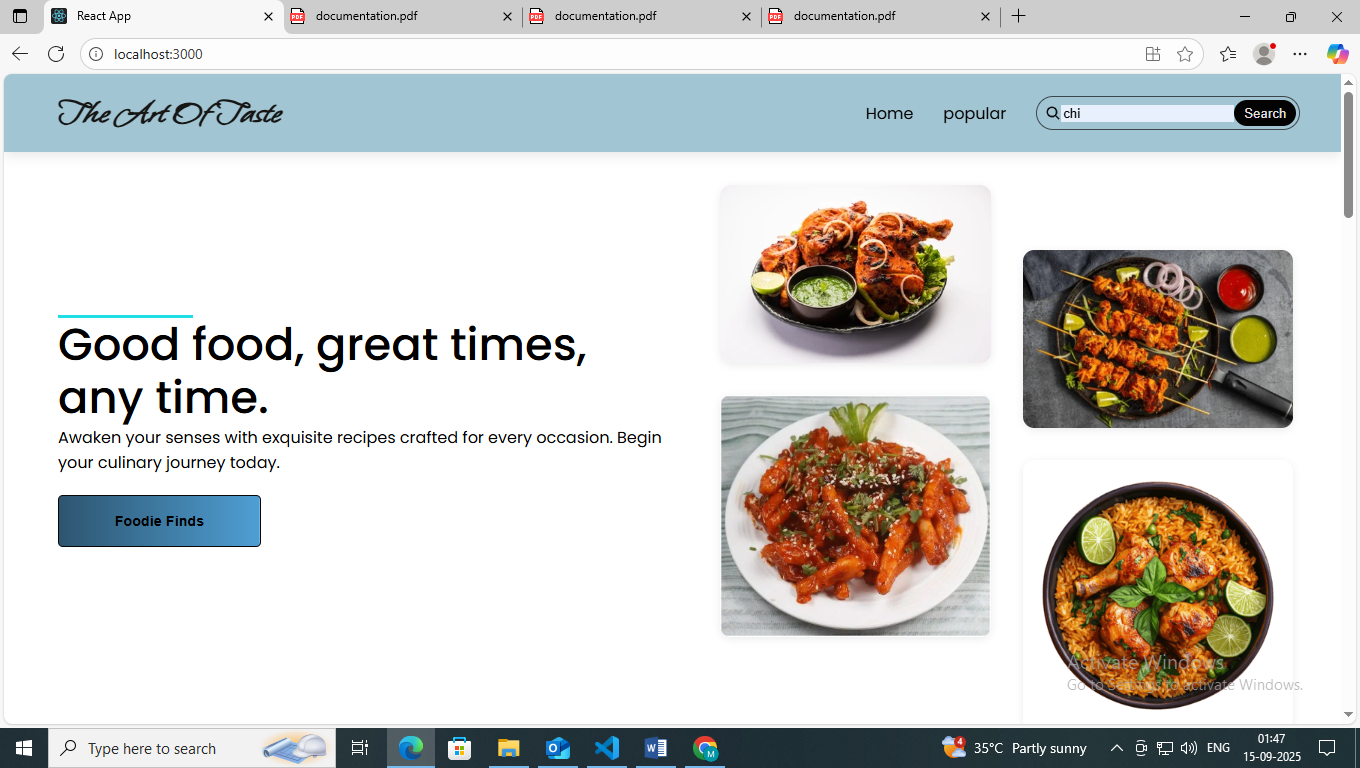
* ***Run tests:***
* *npm test*
* ***Generate coverage report:***

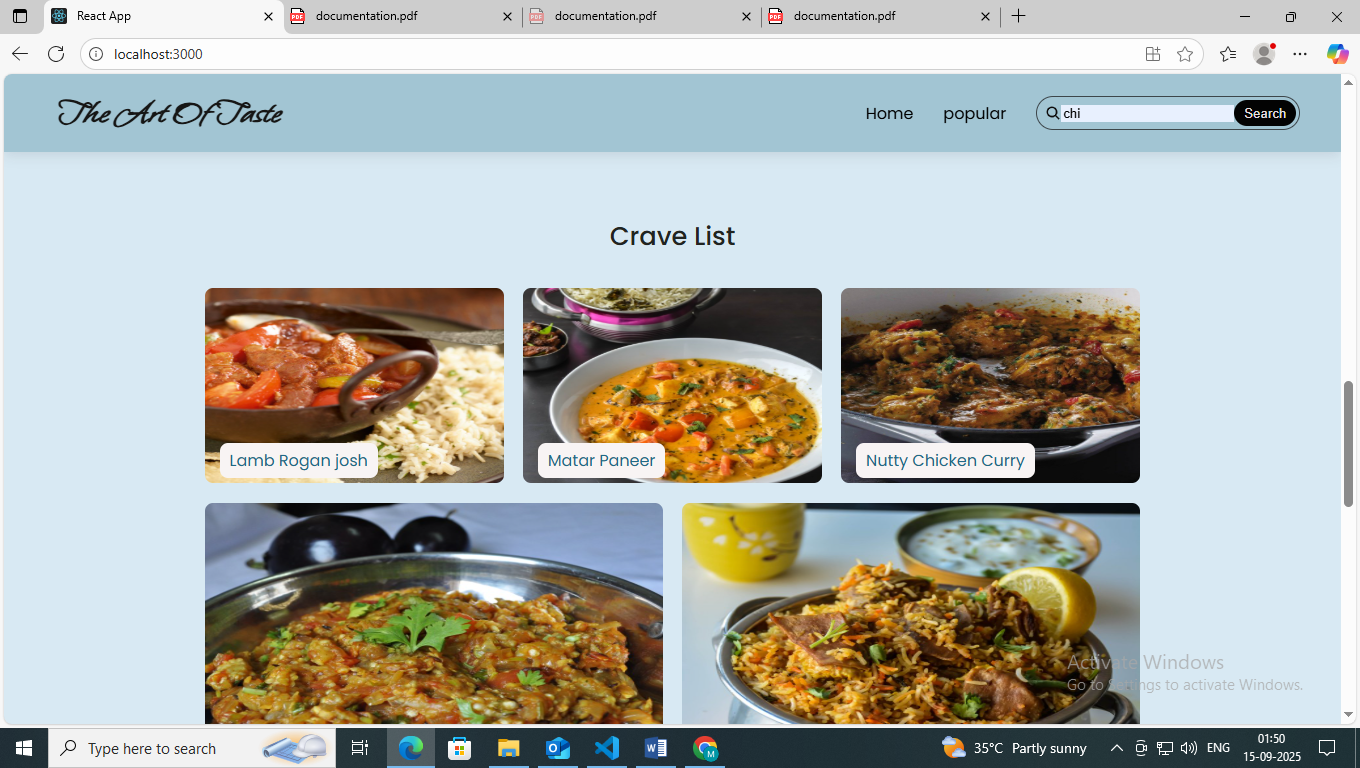
*npm test -- --coverage*

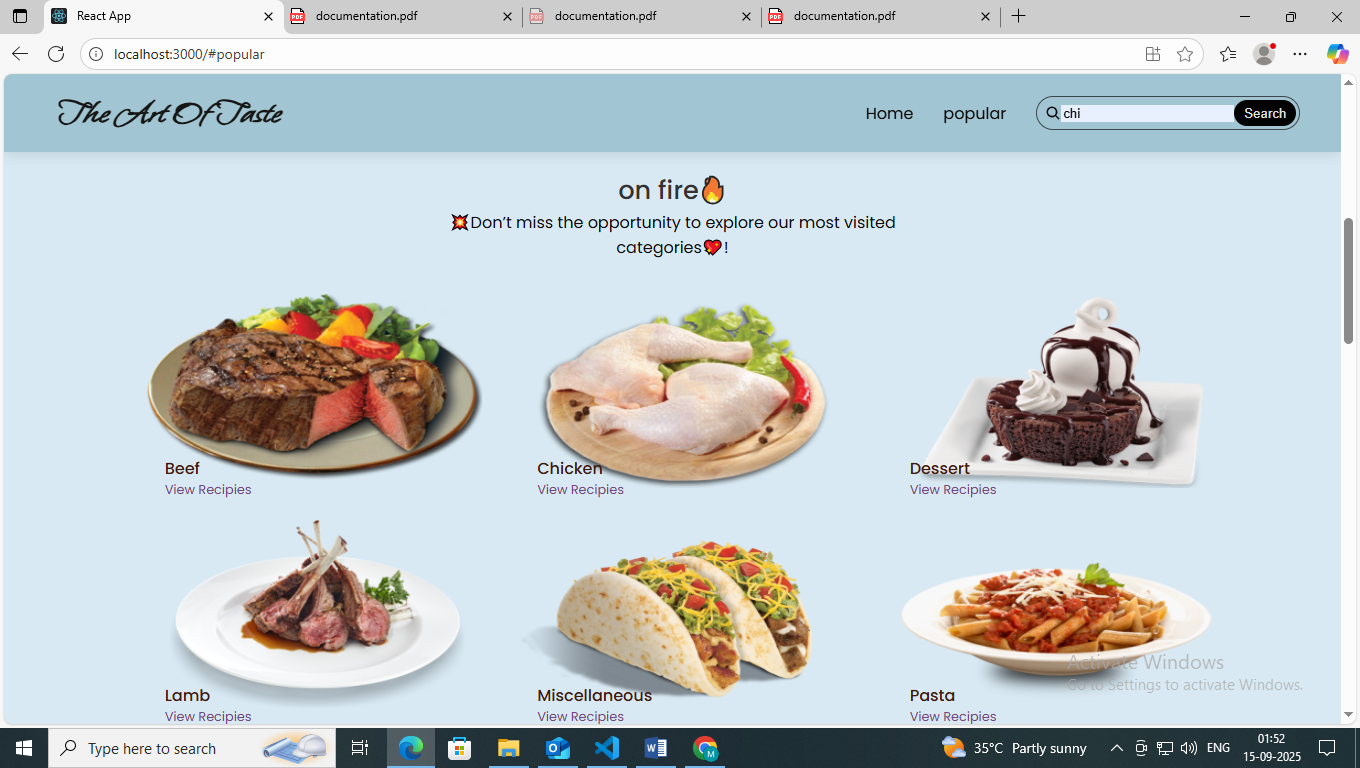
***12. Screenshots or Demo :***

***\* Application Screenshots***

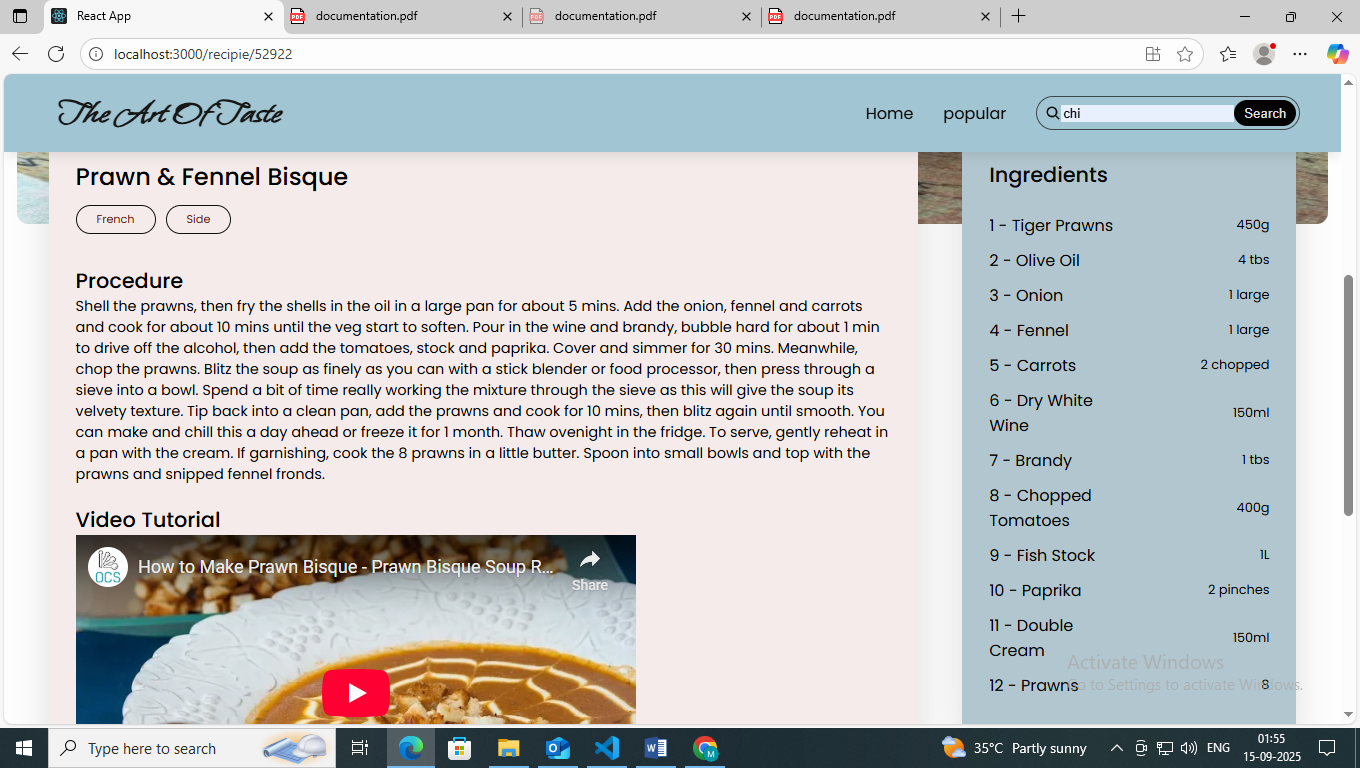
        ***Home Page / Recipe List***

***crave list***

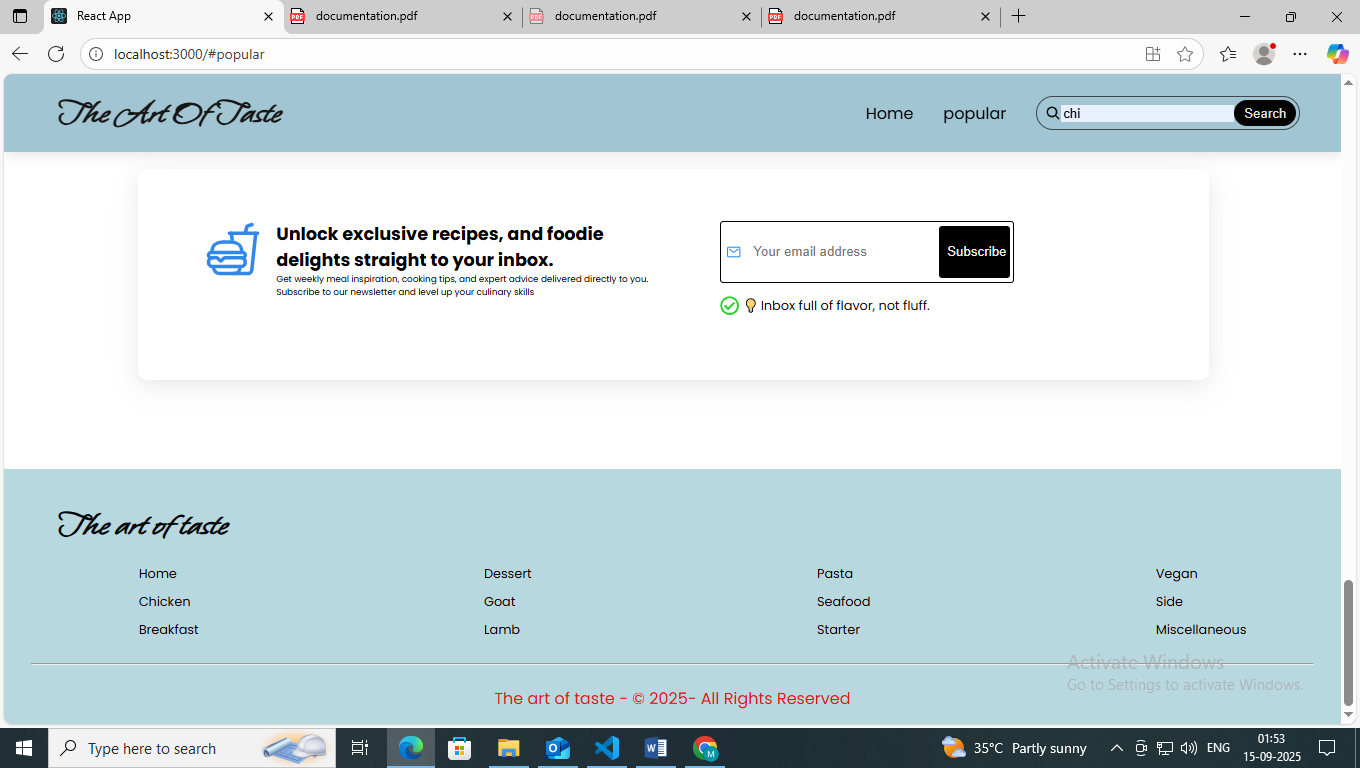


*popular*

*procedure*



*To subscribe*



*Detailed view of a selected recipe with ingredients and instructions.*

***->Live Demo***

*You can also try the application live at:*

<https://drive.google.com/file/d/1lTd3evp5hduAdDJRntCktWXI2T1c9aGj/view?usp=drivesdk>***13. Known Issues :***

* ***Image Upload Limitation:*** *Currently, image uploads for recipes are limited to JPEG and PNG formats only. Uploading other formats may result in errors.*
* ***Form Validation Edge Cases:*** *Some edge cases in the Add/Edit Recipe forms are not fully validated, such as submitting empty ingredient steps or extremely long text inputs, which might cause UI glitches.*
* ***Performance on Large Recipe Lists:*** *When the cookbook contains a number of recipes (50+), the recipe list page may experience slow rendering or lag due to lack of pagination or virtualization.*
* ***Edit Recipe State Persistence:*** *If a user navigates away from the Edit Recipe form without saving, unsaved changes are lost without a warning.*
* ***Mobile Responsiveness:*** *Certain UI elements, such as modal dialogs and forms, may not display optimally on very small screens (e.g., older smartphones).*
* ***Backend API Rate Limiting:*** *The backend does not currently implement API rate limiting, which may lead to unexpected errors if the frontend makes rapid consecutive requests.*

***14. Future Enhancements :***

* ***User Authentication & Profiles:*** *Implement user accounts to allow personalized recipe collections, favorites, and profile management.*
* ***Recipe Ratings & Reviews:*** *Add functionality for users to rate recipes and leave comments or reviews.*
* ***Search and Filtering:*** *Enhance the recipe list with powerful search capabilities and filters by cuisine, difficulty, preparation time, or dietary preferences.*
* ***Pagination and Infinite Scroll:*** *Improve performance on large recipe lists by adding pagination or infinite scrolling.*
* ***Image Gallery & Upload Enhancements:*** *Support multiple images per recipe with an image gallery and drag-and-drop upload functionality.*
* ***Mobile App Version:*** *Develop a native or hybrid mobile app for iOS and Android for better accessibility on mobile devices.*
* ***Offline Support & Data Sync:*** *Enable offline access to saved recipes with background syncing once the device reconnects to the internet.*
* ***Ingredient Auto-Suggestions:*** *Integrate ingredient auto-completion and suggestions while adding or editing recipes for faster data entry.*
* ***Recipe Sharing:*** *Allow users to share recipes via social media, email, or export PDF.*