FRONTEND DEVELOPMENT WITH REACT.JS

COOK BOOK: Your Virtual Kitchen Assistant

1**) Introduction**

* Project Title: COOK BOOK– Your Virtual Kitchen Assistant
* Team Members:
* Anusha . A – Coding and Debugging
* Roshni S.K.– Editing
* Varshini .T – Documentation
* Naveena .M– Demonstration
* Team ID – NM2025TMID39395

2) **Project Overview**

* Purpose
* The cookbook project aims to provide readers with an easy and organized way to explore, prepare, and enjoy a wide variety of recipes.
* It serves to compile different cuisines and cooking styles, enabling quick access to instructions and ingredients as needed.
* The cookbook helps home cooks and food enthusiasts by presenting step-by-step guidance, making cooking both enjoyable and approachable.
* It encourages creativity in the kitchen while ensuring accuracy in ingredient measurement and cooking methods.
* An organized collection of recipes prevents confusion and saves time, helping readers avoid errors and achieve delicious results consistently.
* Goals
* To inspire readers with a diverse selection of recipes.
* To improve cooking skills and kitchen confidence.
* To reduce food waste by offering practical cooking tips and ingredient substitutions.
* To provide a reliable culinary resource that leads to enjoyable meals and satisfied diners.
* Features
* Dashboard views: Centralized overview showing recipe categories, recently added dishes, and cooking trends for easy access.
* Add, update, delete recipes: Simple forms and tables for quick editing, enabling seamless entry, modification, and removal of recipes.
* Image/QR code support: Option to attach food images or QR codes for fast recipe identification and easy sharing.
* Real-time updates: Instant changes when new recipes are added, ingredients are modified, or cooking notes are updated, keeping data accurate and current.
* Multi-cuisine management: Lets users organize and view recipes across different cuisines (e.g., Indian, Italian, Chinese) from one interface.
* Automated suggestions: Smart recommendations based on available ingredients, favorite dishes, or frequently cooked meals.
* Role-based custom views: Different interfaces for chefs, home cooks, or admins, ensuring usability and security.
* Functionality
* Recipe management: Creates, organizes, and tracks recipes with details like ingredients, steps, and cooking times for easy kitchen planning.
* Reporting and analytics: Provides downloadable reports (e.g., CSV exports), detailed insights into cooking trends, and ingredient usage metrics for data-driven meal decisions.
* Ingredient tracking: Records pantry stock levels, substitutions, and restocking needs, with real-time visibility for accurate meal preparation.
* Search and filter options: Advanced querying to quickly locate recipes, sort by cuisine, difficulty, or dietary preference, and optimize recipe discovery.

1. Architecture

* Components Structure
* App.js: Root component, sets up router and context providers.
* Pages: Each distinct view is a separate page (e.g., Home, AddRecipe, Categories, CookingHistory, MealPlanner, Login).
* Components: Reusable UI elements such as Navbar, SearchBar (with mic), RecipeCard, IngredientList, etc.
* Interaction: Pages are composed of these components, passing props for recipe data and handling cooking-related events.
* State Management
* Global State: Managed via React Context API, for features like user authentication, saved recipes, and meal plans.
* Local State: Managed in each component using the use State hook, for form fields (e.g., adding a recipe), modal visibility, cooking timers, etc.
* Routing
* Library Used: react-router-dom
* Structure: Routes are defined in App.js using <Routes> and <Route>. Navigation is handled using <Link> components for client-side routing.

4**) Setup Instructions**

* Prerequisite

Here are the key prerequisites for developing a cookbook frontend application using React.js:



* Node.js and npm:

Node.js is a JavaScript runtime environment that allows you to run code locally. It is essential for building and running the cookbook application. Install Node.js and npm on your machine.

a) Download: https://nodejs.org/en/download/

b) Installation instructions: https://nodejs.org/en/download/package-manager/

* React.js:

React.js is used to build the interactive cookbook UI, including recipe pages, ingredient lists, and meal planners.

a) Create a new React app:

I. npx create-react-app my-cookbook-app

ii. Replace my-cookbook-app with your preferred project name.

b) Navigate to the project directory: cd my-cookbook-app.

c) Running the Cookbook App: Once created, start the development server and view your

cookbook in the browser.

d) Start the development server: npm start

This launches the server, and you can access your cookbook app at http://localhost:3000.

* HTML, CSS, and JavaScript:
* Basic knowledge is required:
* HTML for structuring recipe pages.
* CSS for styling recipe cards, ingredient lists, and layouts.
* JavaScript for adding interactivity (e.g., timers, search, filters).
* Version Control (Git):

Use Git to track recipe app changes and collaborate. Hosting platforms like GitHub or Bitbucket can manage your project repository.

Git download: https://git-scm.com/downloads

* Development Environment:

Choose an IDE or code editor to work on the cookbook project. Recommended options:

* Visual Studio Code: <https://code.visualstudio.com/download>
* Sublime Text: <https://www.sublimetext.com/download>
* WebStorm: <https://www.jetbrains.com/webstorm/download>
* To get the Cookbook App project from drive:

Follow these steps:

* Install Dependencies:

Navigate into the cloned cookbook repository directory and install libraries:

* + 1. cd cookbook
    2. npm install
* Start the Development Server:

Run the following command: npm start

* Access the Cookbook App:

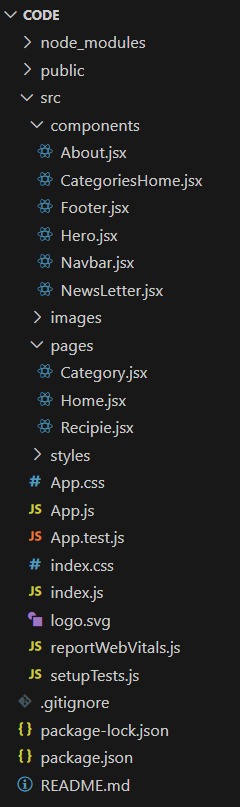
Open your browser and go to: http://localhost:3000

You should see the cookbook homepage with recipe categories and sample dishes.

✅ You have successfully installed and set up the Cookbook Application on your local machine. You can now begin adding recipes, customizing layouts, and testing features like ingredient tracking and meal planning.

5) **Folder Structure**

The image is of the folder structure which shows all the files and folders that have been used in project development.



6) **Running the Applications**

Start the frontend server locally:

* Navigate to the client directory: cd client
* Run: npm start

The application is available at http://localhost:3000.

7**) Components Documentation**

➢ Key Components

a) Navbar: Displays navigation and user info.

b) CategoryCard: Renders a category with image and name; receives category, image, and

onSelect props.

c) Search Bar: Input field with search and mic icon; receives on Search, placeholder props.

➢ Reusable Components

a) Button: Configurable for style and event.

b) Modal: Used for popups, configurable via props.

c) Input Field: For text input, props for value and on Change.

8**) State Management**

➢ Global State: Managed using React Context (e.g., User Context for authentication, Cart Context for cart), provided at the top level so all components can access global state.

➢ Local State: Handled by use State within components, for things like form fields, toggles, and transient UI state.

9**) User Interface**

* Home: The main landing page, featuring the most popular categories and recipes.
* Most Popular Categories: A section that highlights the most frequently viewed or trending recipe categories.
* Recipes: A dedicated page or section that lists all available recipes.
* Search Bar: An input field that allows users to search for specific recipes or ingredients.
* Inbox: A section for user messages, notifications, or a newsletter.
* Footer: The bottom section of the page, containing navigation links or contact information.

**10) Styling**

➢ CSS Frameworks / Libraries

a) Custom CSS (Styles.css) used for most UI.

b) No third-party frameworks (e.g., Bootstrap) unless specified.

c) Optionally, Styled-Components for scoped styling.

➢ Theming

a) Color variables and CSS custom properties enable easy theme changes

(light/dark modes).

11**) Testing**

* Testing Strategy

a) Unit Tests: For components and utilities using Jest and React Testing Library.

b) Integration Tests: For flows like login and cart.

c) End-to-End Tests: (Optional) Can use Cypress.

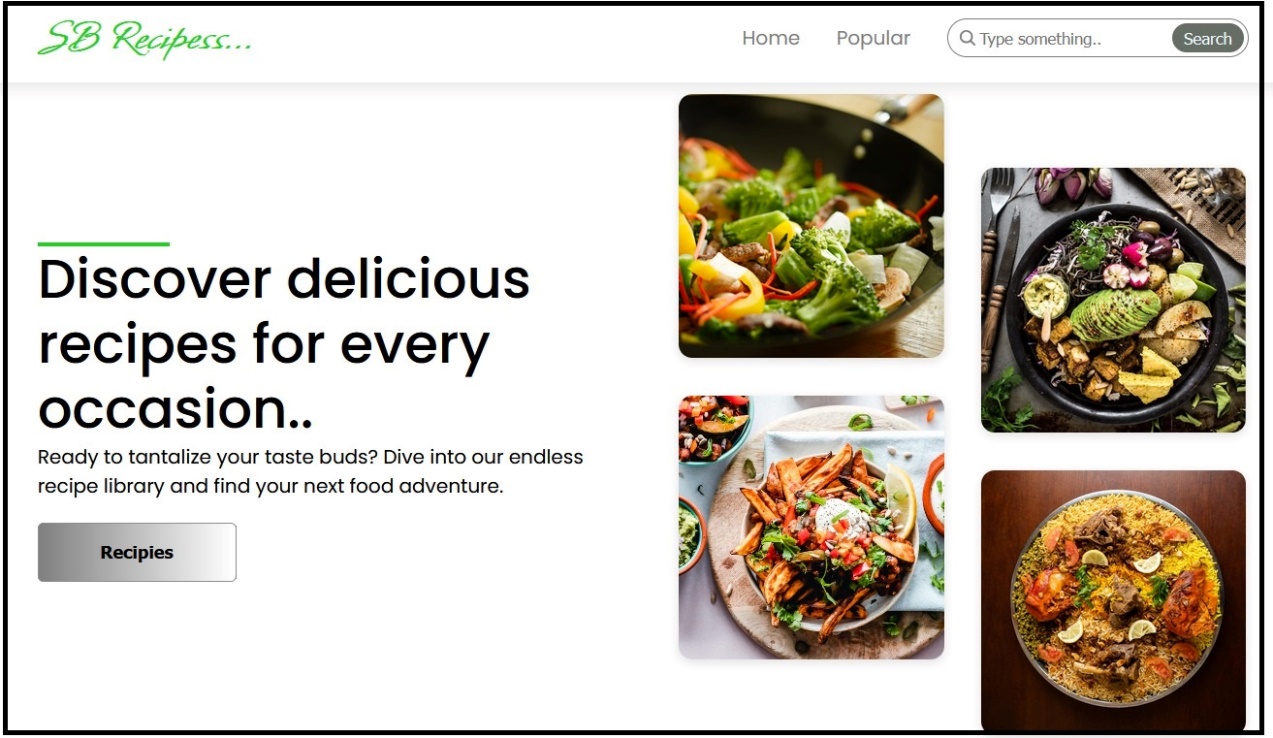
* Code Coverage

a) Run npm test -- --coverage to generate coverage reports.

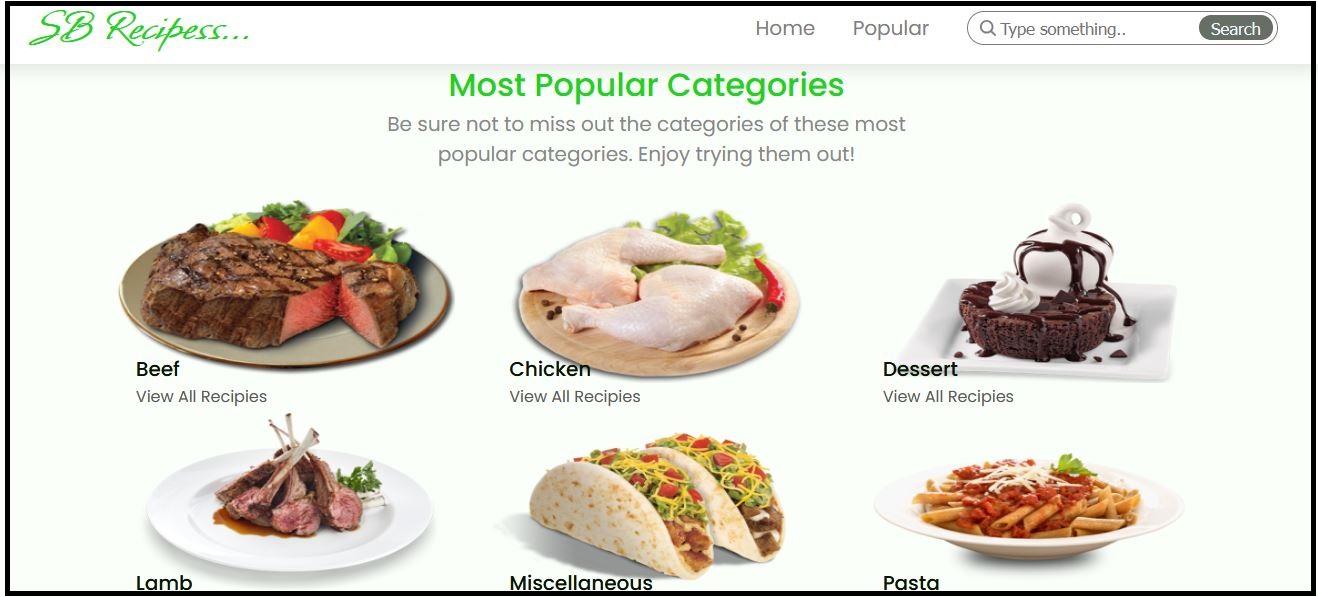
12**) Screenshots and Demo**

➢ Screenshots

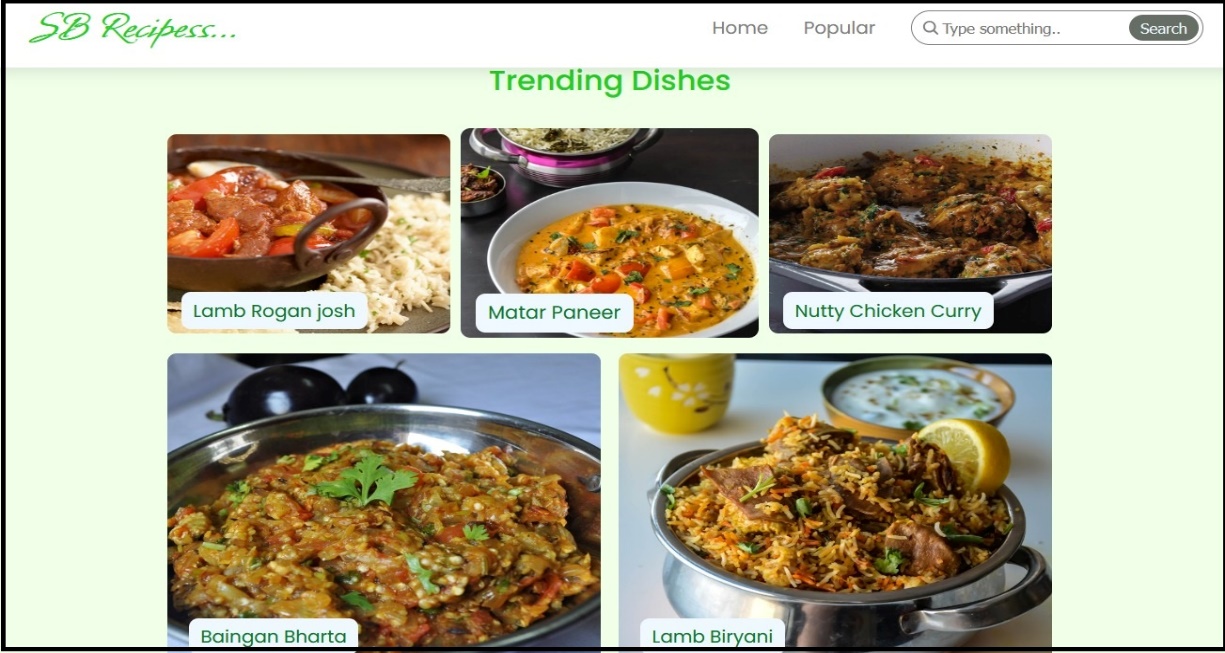
* **Home**



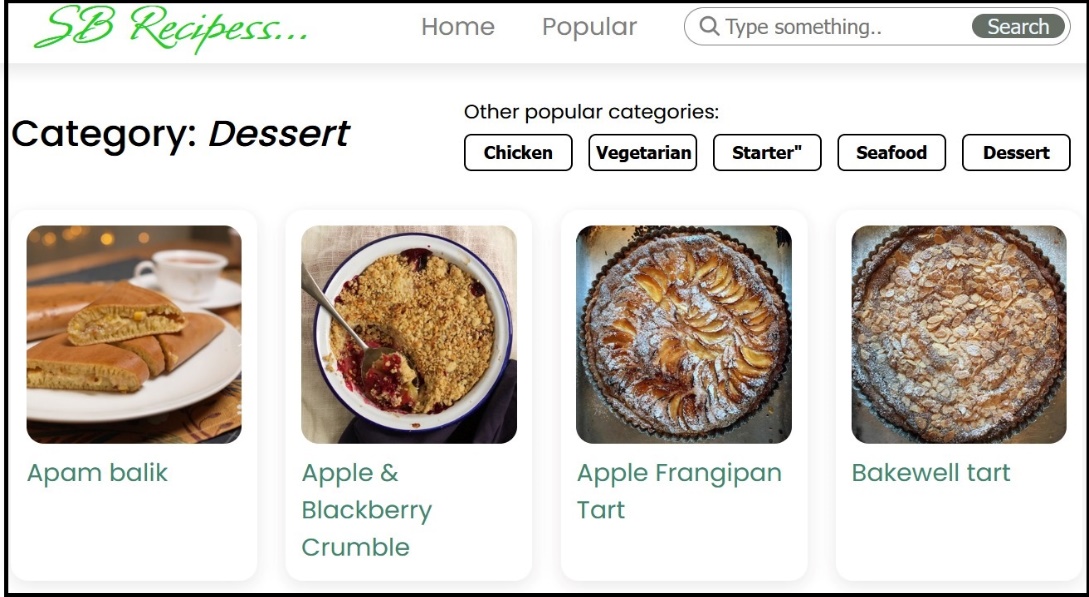
* **Most popular Categories**



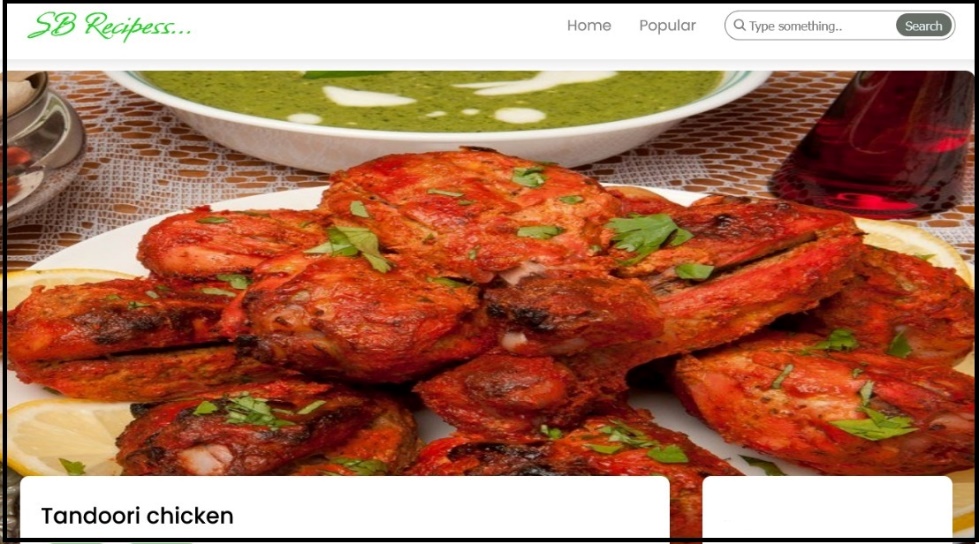
* **Trending Dishes**



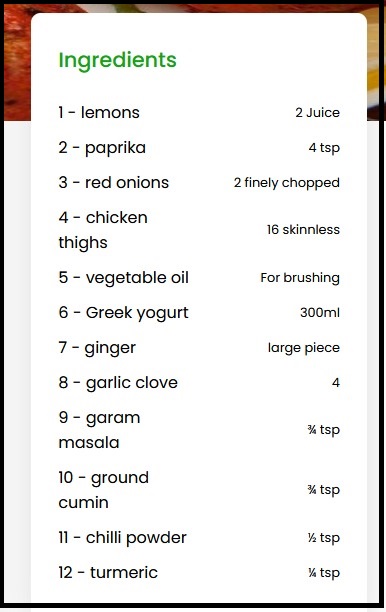
* **Recipes**



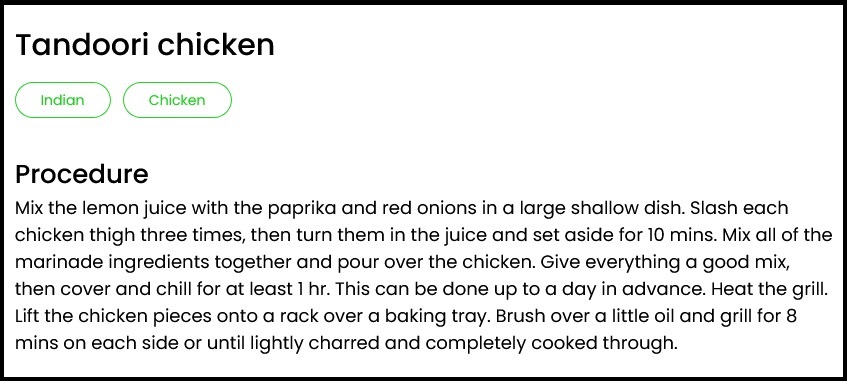
* **Dishes**



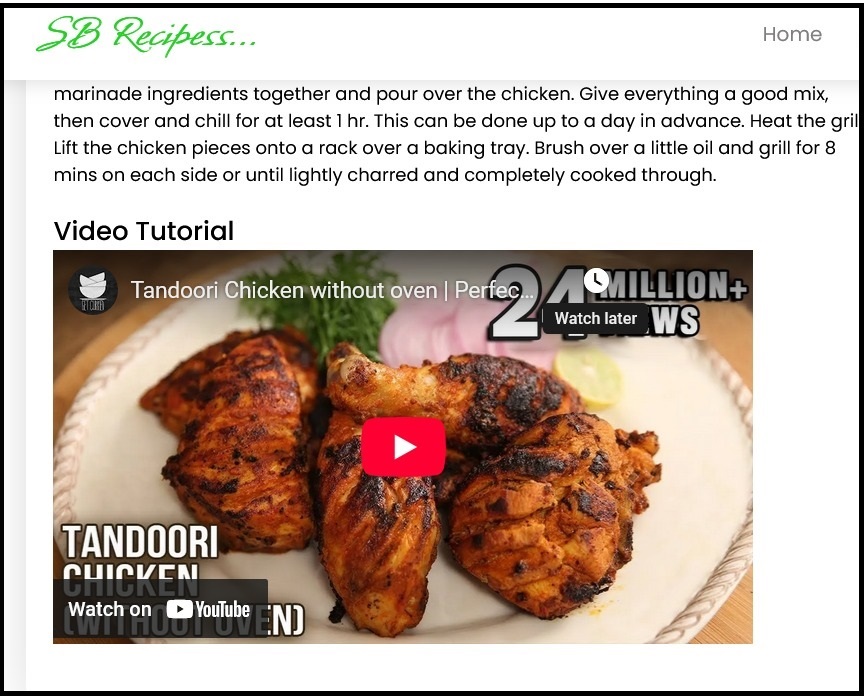
* **Ingredients**



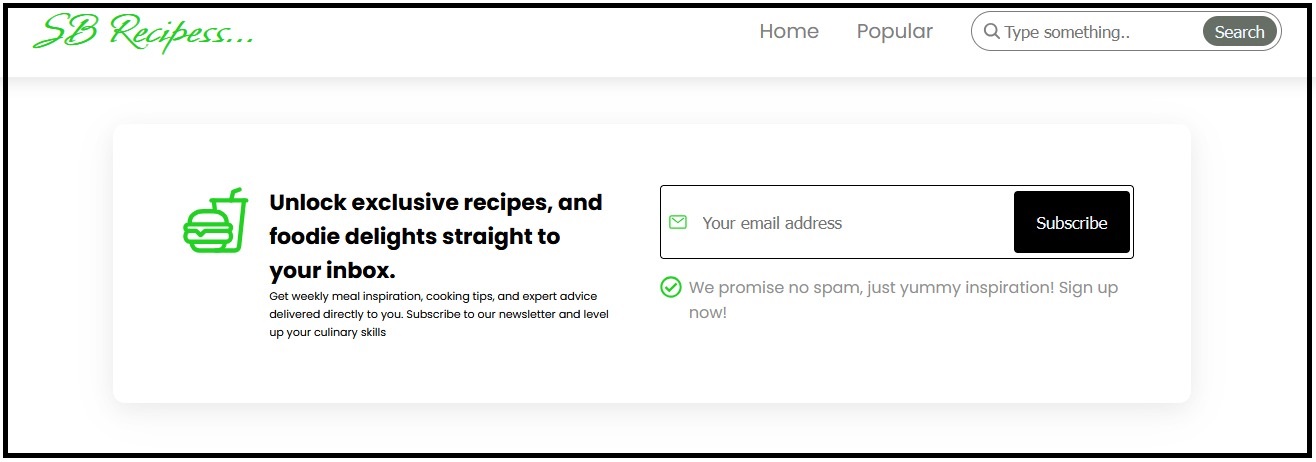
* **Procedure**



* **Video Tutorial**



* **Inbox**



* **Footer**



➢ Demo video Link:

<https://drive.google.com/file/d/17ypJXDe6f0Ln5kxR5p9YOhflDVwSq5dH/view>

13) **Known Issues**

* Speech recognition (mic) works only in Chrome/Edge, not Firefox/Safari.
* Mobile responsiveness needs improvement on some pages.
* Unsplash category images may not load offline.

14) **Future Enhancements**

* Backend integration for real-time recipe updates.
* Improved mobile and responsive design for all devices.
* Progressive Web App (PWA) features for offline access and push notifications.
* More UI animations and transitions for a fluid user experience.
* Advanced search and filtering by dietary needs, ingredients, or cooking time.
* Enhanced theming and user customization to personalize the app's look.