COOKBOOK

Your Virtual kitchen assistant

(React Application)

A fusion of flavors and recipes, perfect for a dynamic cooking experience.

DR,MGR JANAKI COLLEGE OF ARTS AND SCIENCE FOR WOMEN

(B.Sc., Computer Science-Final Year)

Presented By:

Pavithra.M (asunm1423222208014)

Syedali Fathima.K (asunm1423222208025)

Jeevashankari.R (asunm1423222207997)

Virundha.V (asunm1423222208031)

2. Project Overview

Purpose:

A fusion of flavors and creativity, perfect for a delightful culinary journey with CookBook: Your Virtual Kitchen Assistant.

The primary purpose of the **CookBook: Your Virtual Kitchen Assistant** is to provide an interactive and user-friendly platform for food enthusiasts to explore, share, and discover recipes online. This React-based web application serves as a virtual kitchen assistant that simplifies the cooking journey by offering a seamless recipe browsing experience.

Goals:

- To create an intuitive interface for users to browse a wide range of recipes based on categories.
- To allow users to search and view detailed recipes with ingredients and preparation instructions.
- To provide a visually appealing design that enhances user experience.
- To categorize recipes into different meal types such as Breakfast, Lunch, Dinner, Desserts, and Beverages.
- To offer responsive design compatibility for both desktop and mobile devices.
- To make the platform user-friendly, lightweight, and easy to navigate.
- To establish the foundation for future enhancements like user authentication, recipe uploading, and saving favorite recipes.

Features:

- **Home Page:** Welcoming interface with an overview of popular recipes and categories.
- **Recipe Page:** Detailed view of recipes with images, ingredients, and step-by-step instructions.
- **Categories Page:** Organized recipe categories for quick browsing.
- **Search Functionality:** Quick search to find recipes by name or ingredients.
- **Responsive Design:** Fully optimized for desktop, tablet, and mobile devices.
- **Navigation Bar:** Seamless navigation between pages.
- **About Page:** Information about the platform and its purpose.
- **Contact Page:** Easy-to-access form for user inquiries or feedback.

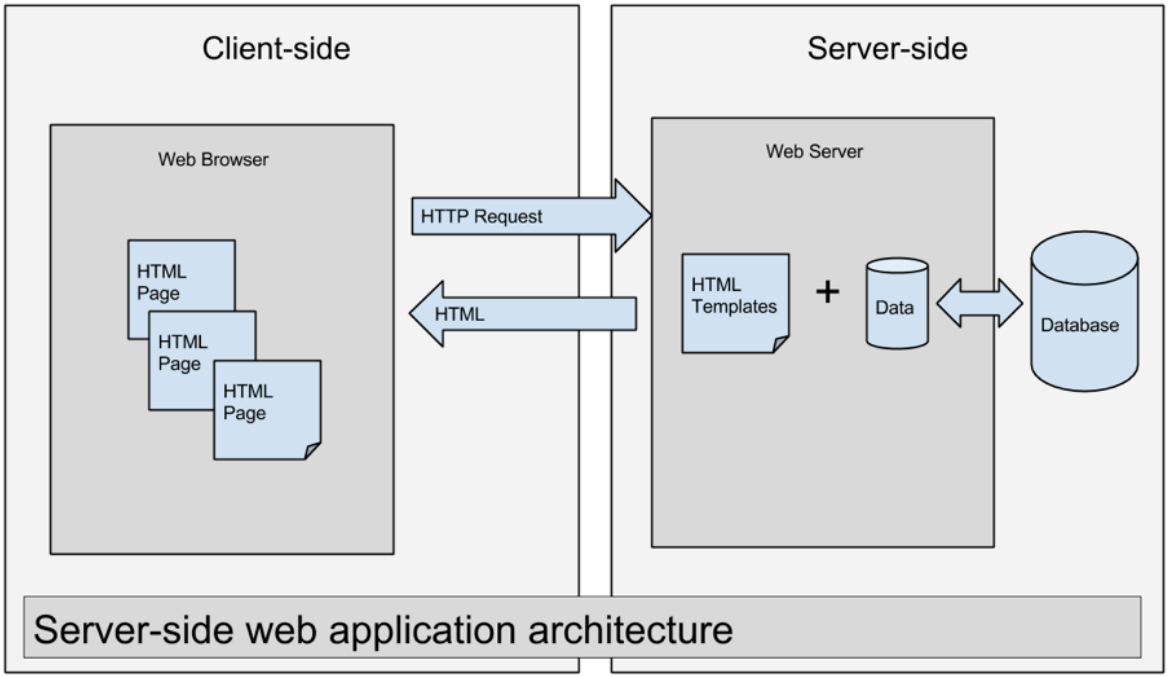
3. Architecture

Component Structure:

The **CookBook** application follows a modular component-based architecture in React, ensuring scalability and reusability. The key components and their interactions are outlined below:

- **App Component:** The root component that manages routing and serves as the main container.
- **Navbar Component:** Handles navigation across all pages.
- **Home Component:** Displays featured recipes and categories.
- **RecipeList Component:** Renders a list of recipes based on selected categories or search queries.
- **RecipeCard Component:** Shows individual recipe details with image, title, and a link to the full recipe.
- **CategoryCard Component:** Represents different recipe categories with images.
- **About Component:** Provides information about the platform.
- **Contact Component:** Displays the contact form.
- **Footer Component:** Contains website footer information.

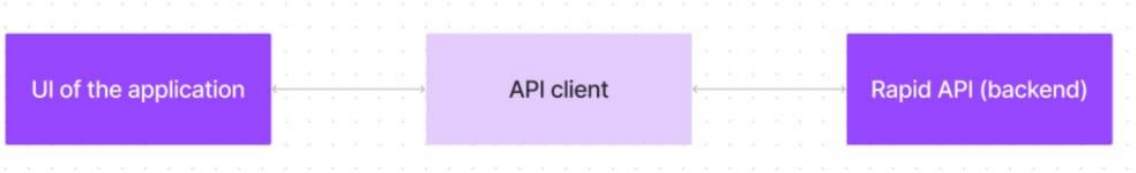
The components are organized in a hierarchical structure, ensuring smooth data flow and user experience.



State Management:

The **CookBook** application uses **React's built-in useState and useEffect hooks** for state management. The state is primarily managed at the component level to handle user interactions, dynamic content rendering, and data fetching.

- **useState:** Manages component-level state like search input, selected categories, and form data.
- **useEffect:** Handles side effects like fetching recipe data and updating components based on user actions.
- **Props Drilling:** Data is passed between parent and child components using props to ensure consistent information flow.



Routing:

The **CookBook** application uses **React Router** to manage client-side navigation between different pages. The routing structure ensures seamless transitions and a dynamic browsing experience.

- **BrowserRouter:** Wraps the entire application to enable routing.
- **Routes Component:** Defines all application routes.
- **Route Component:** Maps different URLs to their respective components such as Home, Recipes, Categories, About, and Contact.
- **useNavigate Hook:** Allows programmatic navigation between pages.
- **NavLink Component:** Used for navigation links with active link highlighting.

```
<Routes>

 <Route path="/" element={<Home />} />
 <Route path="/category/:id" element={<Category />} />
 <Route path="/recipie/:id" element={<Recipie />} />
</Routes>
```

4. Setup Instructions

Prerequisites:

Before setting up the **CookBook** application, ensure that the following software dependencies are installed on your system:

- **Node.js (v14 or later):** Required for running the development server and managing dependencies.
- **npm (Node Package Manager):** Comes bundled with Node.js to install project dependencies.
- **Git:** For cloning the project repository.

- **Code Editor:** Recommended **Visual Studio Code** for a better development experience.
- **Browser:** Google Chrome or any modern browser to test the application.

Installation:

Follow these steps to set up the **CookBook** application:

1. **Clone the Repository:**
2. git clone https://github.com/username/cookbook.git

cd cookbook

3. **Install Dependencies:**

npm install

4. **Configure Environment Variables:** Create a .env file in the project root directory and add the following environment variables:
5. REACT_APP_API_URL=https://api.example.com

REACT_APP_SITE_NAME=CookBook

6. **Run the Development Server:**

npm run dev

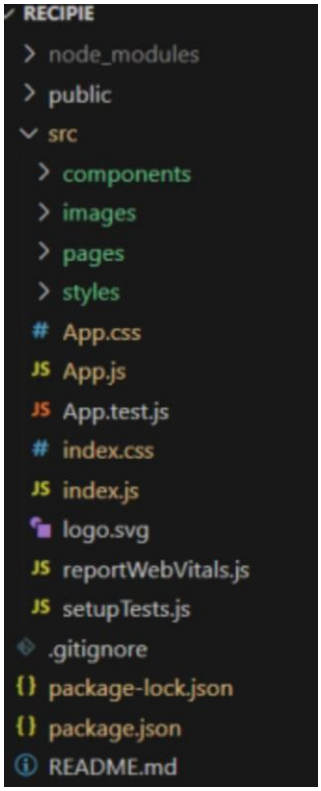
7. **Access the Application:** Open your browser and navigate to http://localhost:3000 to view the application.

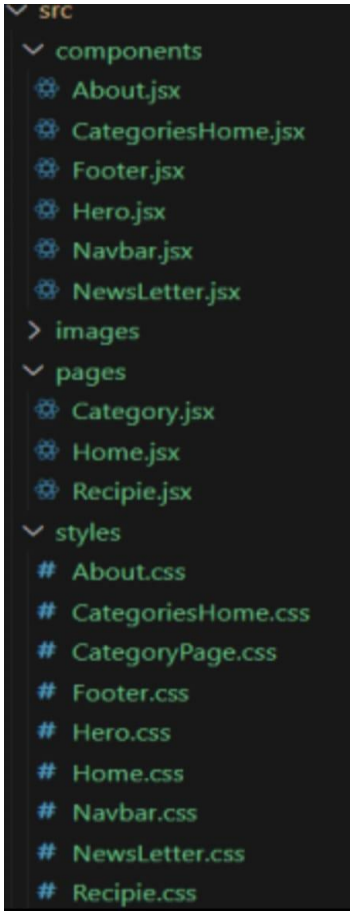
5. Folder Structure

Client:

The **CookBook** application's React project is organized into the following folder structure to maintain clarity and separation of concerns:

- **src/**: Main source code folder containing all application files.
    - **components/**: Reusable UI components like Navbar, Footer, and Cards.
    - **pages/**: Page components such as Home, Recipes, Categories, About, and Contact.
    - **assets/**: Static assets like images, icons, and CSS files.
    - **hooks/**: Custom hooks to handle reusable logic.
    - **utils/**: Utility functions used throughout the application.
    - **App.jsx:** Main application component.
    - **main.jsx:** Application entry point.
    - **router.jsx:** Defines application routes.





Utilities:

The **CookBook** application utilizes custom hooks and utility functions to streamline reusable logic across components.

- **Custom Hooks:**
    - useFetchRecipes: Custom hook to fetch recipe data from an API.
    - useWindowWidth: Hook to get the current window width for responsive components.
- **Utility Functions:**
    - formatDate(date): Formats date strings into a readable format.
    - filterRecipes(recipes, query): Filters recipe list based on search queries.
    - capitalizeFirstLetter(string): Capitalizes the first letter of a string.

6. Running the Application

To run the application locally, follow these steps:

- **Frontend**
  Navigate to the project directory where your React application is located.

Use the following commands:

bash

CopyEdit

Go to the project folder

cd cookbook

Install dependencies

npm install

Start the frontend server

npm run dev

7. Component Documentation

This section describes the **key components** of **CookBook: Your Virtual Kitchen Assistant** built with React, their **purpose**, and the **props** they receive.

1. Navbar.jsx

- **Purpose:** Displays the navigation bar with links to different pages.
- **Props:**
    - title (optional) – Title of the website (default: **CookBook**)

---

2. Home.jsx

- **Purpose:** Serves as the homepage with a welcome message and featured recipes.
- **Props:**
    - None

---

3. Recipes.jsx

- **Purpose:** Displays the list of recipes with images, descriptions, and links.
- **Props:**
    - recipes – Array of recipe objects with properties like title, image, and description.

---

4. Categories.jsx

- **Purpose:** Showcases different recipe categories (e.g., Breakfast, Lunch, Dessert).
- **Props:**
    - categories – Array of category objects with properties like name, image, and id.

---

5. About.jsx

- **Purpose:** Provides information about the website and its purpose.
- **Props:**
    - None

---

6. Contact.jsx

- **Purpose:** Displays the contact form with fields like name, email, and message.
- **Props:**
    - None

---

7. Footer.jsx

- **Purpose:** Displays the website's footer with social media links and copyright information.

- **Props:**
    - year – Current year (dynamic)

Reusable Components

This section lists the **Reusable Components** used in **CookBook: Your Virtual Kitchen Assistant** with a simple explanation.

---

1. Button.jsx

- **Purpose:** Used for clickable buttons.
- **Props:**
    - text – Button text.
    - onClick – Action when clicked.

Example Usage:

jsx

CopyEdit

<Button text="Explore Recipes" onClick={handleClick} />

---

2. Card.jsx

- **Purpose:** Displays recipes or categories with image and title.
- **Props:**
    - title – Card title.
    - image – Image URL.

Example Usage:

jsx

CopyEdit

<Card title="Pasta" image="/images/pasta.jpg" />

---

3. Input.jsx

- **Purpose:** Used for input fields in forms.
- **Props:**
    - type – Text or Email.
    - placeholder – Placeholder text.

Example Usage:

jsx

CopyEdit

<Input type="text" placeholder="Enter Name" />

---

4. Loader.jsx

- **Purpose:** Shows loading spinner.
- **Props:**
    - None

Example Usage:

jsx

CopyEdit

{loading ? <Loader /> : <Recipes />}

8. State Management

This section explains how **state management** is handled in **CookBook: Your Virtual Kitchen Assistant**.

---

1. Global State

- **Purpose:** Stores data that needs to be shared across multiple components.
- **Implementation:** Managed using **React Context API**.
- **Example Usage:** Global state like selected categories or user login status is stored in the App.jsx component and passed down using **props**.

Example:

jsx

CopyEdit

const [selectedCategory, setSelectedCategory] = useState("All");

<CategoryItem name="Desserts" onClick={() => setSelectedCategory("Desserts")} />

---

2. Local State

- **Purpose:** Stores data that is only needed within a single component.
- **Implementation:** Managed using the **useState** hook.

Example: In the **Contact.jsx** component, local state is used to handle form inputs:

jsx

CopyEdit

```
const [name, setName] = useState("");


<Input type="text" placeholder="Enter Name" value={name} onChange={(e) =>
setName(e.target.value)} />
```

9. User Interface

This section showcases the **User Interface** of **CookBook: Your Virtual Kitchen Assistant** with a description of its key features.

---

1. Homepage

- Welcome message with a brief introduction.
- Button to explore recipes.

---

2. Recipes Page

- Grid layout displaying recipe cards.
- Each card contains:
    - Recipe Image
    - Title
    - Short Description

---

3. Categories Page

- Display different recipe categories like:
    - Breakfast
    - Lunch
    - Desserts
- Clickable cards to filter recipes by category.

---

4. About Page

- Short description of the website.
- Purpose of the project.

---

5. Contact Page

- Simple form with:
  - Name
  - Email
  - Message
- Submit button to send user queries.

---

6. Footer

- Social media links.
- Copyright information.

10. Styling

This section explains the **styling techniques** used in **CookBook: Your Virtual Kitchen Assistant**.

---

1. CSS Frameworks/Libraries

- **Framework Used: Tailwind CSS**
- Purpose: For quick and responsive design.
- Usage: Applied for layout, colors, and spacing.

Example Usage:

jsx

CopyEdit

```
<div className="bg-orange-200 text-center p-4">
  Welcome to CookBook!
</div>
```

---

2. Theming

- Custom design system with consistent:
    - Colors: Orange and White
    - Fonts: Sans-serif
    - Buttons with rounded corners

---

3. Responsive Design

- Mobile-friendly layout using **Tailwind CSS**'s built-in breakpoints.
- Example:

jsx

CopyEdit

```
<div className="grid grid-cols-1 md:grid-cols-3 gap-4">
  {/* Cards */}
</div>
```

11. Testing

This section describes the **testing approach** used in **CookBook: Your Virtual Kitchen Assistant**.

---

1. Testing Strategy

Currently, the project follows **manual testing** to check the functionality of each component.

Testing Approach:

- Unit Testing: Testing individual components like **Button** and **Card** to ensure they render correctly.
- Integration Testing: Checking if pages like **Recipes** and **Categories** display the correct content based on data.
- User Interface Testing: Manually verifying forms, navigation, and responsiveness across devices.

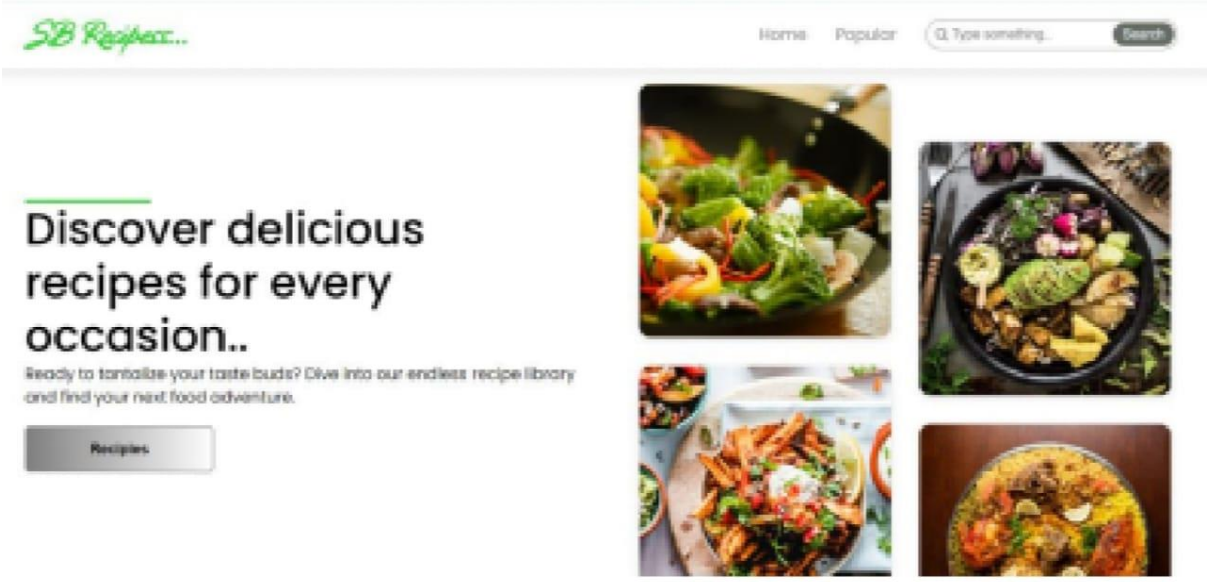
---

2. Code Coverage

- No automated test coverage tools are used in this project.
- Testing is done by:

- Manually checking component rendering.
- Verifying functionality like button clicks and form submissions.

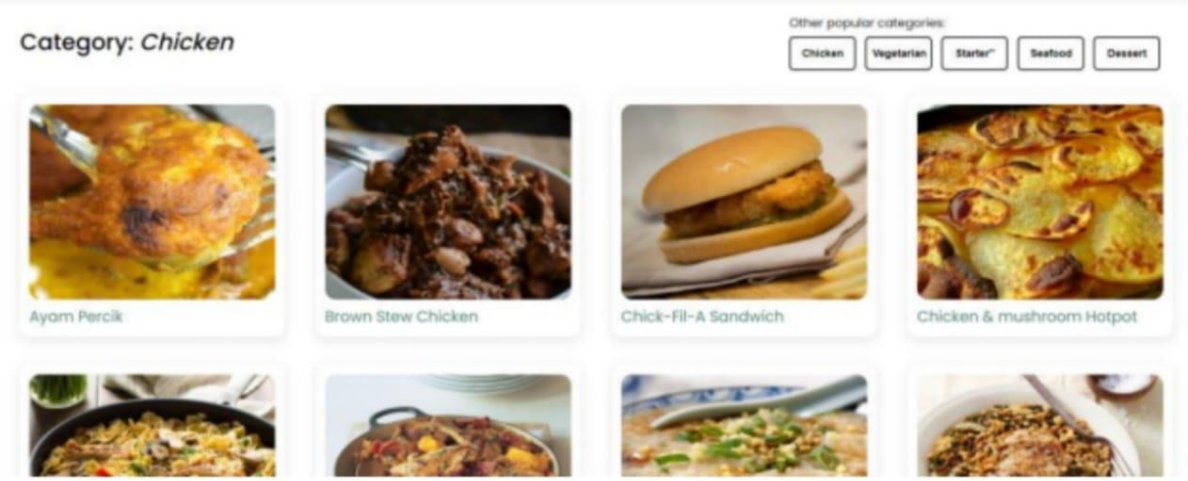
12. **Screenshots**



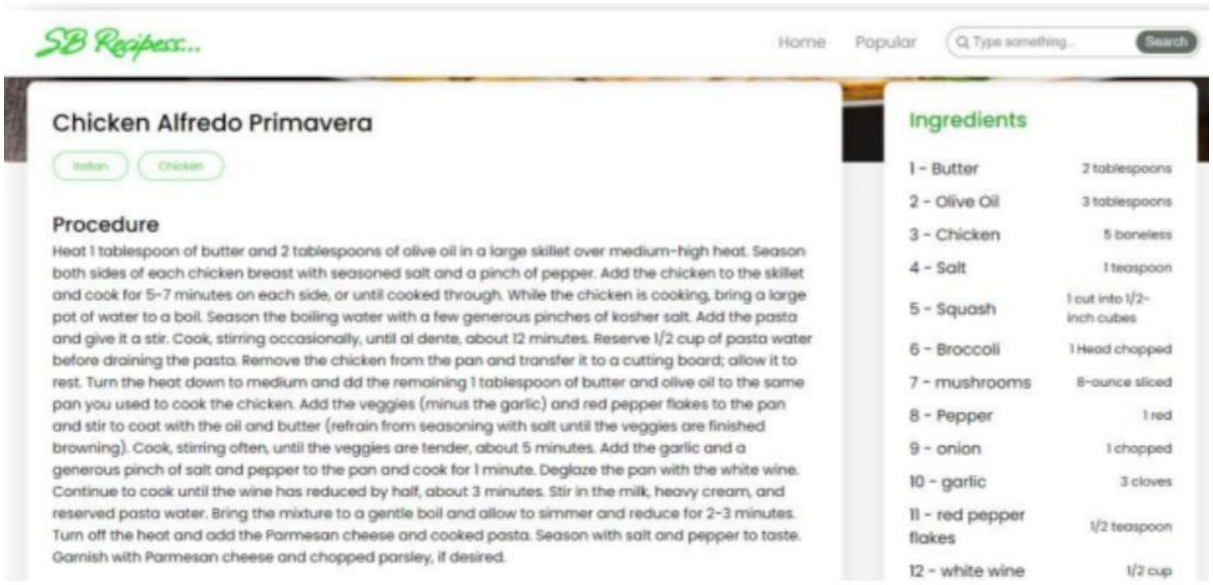
Popular Categories:

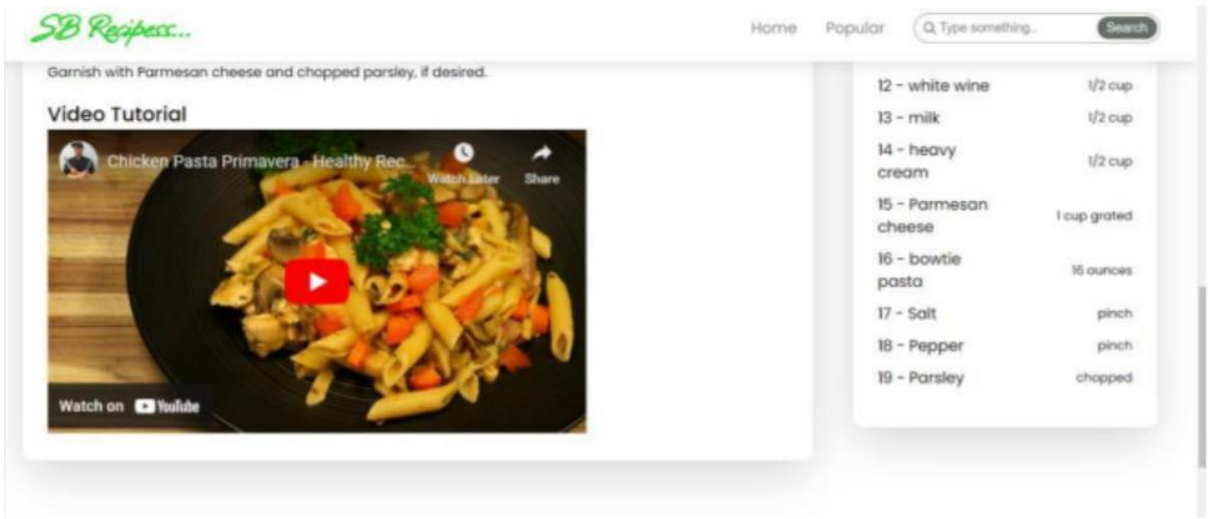
Trending Dishes:

Categories Dishes Page:



Recipe Page:





14. Future Enhancements

Here are some **future features and improvements** planned for **CookBook: Your Virtual Kitchen Assistant**:

1. **Search Functionality** – Allow users to search recipes by name.

2. **Recipe Filtering** – Filter recipes by category or ingredients.
3. **User Login System** – Let users create accounts and save favorite recipes.
4. **Dark Mode** – Add light and dark theme options.
5. **Animations** – Add smooth animations for better user experience.