**Frontend Development with React.js**

**Project Documentation**

1. Introduction

Project Title: Cookbook – Your Virtual Kitchen Assistance

Team Members:

V. Aishwarya

U. Geedhavaani

N. Janani

R. Marshini

2. Project Overview

Purpose:

The project aims to create a virtual kitchen assistant that helps users explore, manage, and cook recipes in a user-friendly way. It provides step-by-step guidance, ingredient tracking, and meal planning features.

Features:

User-friendly recipe search and browsing.

Step-by-step cooking guidance.

Save favorite recipes and meal plans.

Filter recipes by cuisine, ingredients, or diet preferences.

Responsive design for mobile and desktop use.

3. Architecture

Component Structure:

App.js: Main entry point for the React app.

Navbar Component: Provides navigation links.

RecipeList Component: Displays a list of available recipes.

RecipeDetail Component: Shows detailed instructions and ingredients.

Favorites Component: Stores user’s favorite recipes.

Search Component: Allows filtering and searching recipes.

State Management:

Implemented using React Context API for global state.

Local states managed with React hooks (useState, useEffect).

Routing:

Handled with react-router-dom.

Routes include:

/ → Home

/recipes → Recipe List

/recipes/:id → Recipe Detail

/favorites → Favorites

4. Setup Instructions

Prerequisites:

Node.js (>= 16.x)

npm or yarn

Git

Installation:

1. Clone the repository:

git clone https://github.com/your-repo/cookbook-virtual-kitchen.git

2. Navigate to the client folder:

cd cookbook-virtual-kitchen/client

3. Install dependencies:

npm install

4. Configure environment variables (e.g., API keys if needed).

5. Folder Structure

client/

│── public/ # Static files

│── src/

│ ├── assets/ # Images, icons, etc.

│ ├── components/ # Reusable components

│ ├── pages/ # Page-level components (Home, Recipes, Favorites)

│ ├── hooks/ # Custom hooks

│ ├── utils/ # Helper functions

│ ├── App.js # Main app component

│ └── index.js # Entry point

Client: Organized into components, pages, assets, hooks, utils.

Utilities: Includes helper functions for API calls, filtering, and custom hooks (e.g., useFetch).

6. Running the Application

Start the frontend server locally:

npm start

Runs on: http://localhost:3000/

7. Component Documentation

Key Components:

Navbar: Provides navigation across the app.

RecipeList: Displays recipe cards.

RecipeDetail: Shows cooking steps and ingredients.

Favorites: Manages saved recipes.

Reusable Components:

Button Component: Styled button used across pages.

Card Component: Recipe display format.

Modal Component: For popups and quick recipe views.

8. State Management

Global State

The application uses Context API / Redux (choose based on your project) to manage global state.

Examples include:

Storing user preferences (e.g., dietary restrictions, favorite recipes).

Managing authentication and session details.

Sharing recipe data across multiple pages.

State flows from a central provider down to child components, ensuring data consistency across the application.

Local State

Local state is handled using React’s useState and useReducer hooks.

Examples include:

Handling form inputs (search bar, ingredient entry).

Controlling UI interactions (modals, dropdowns, step-by-step recipe progress).

Managing temporary UI states (loading indicators, error messages).

9.User Interface

Overview

Cookbook provides an interactive kitchen assistant experience with recipe browsing, ingredient-based search, and personalized cooking suggestions.

Key UI Features

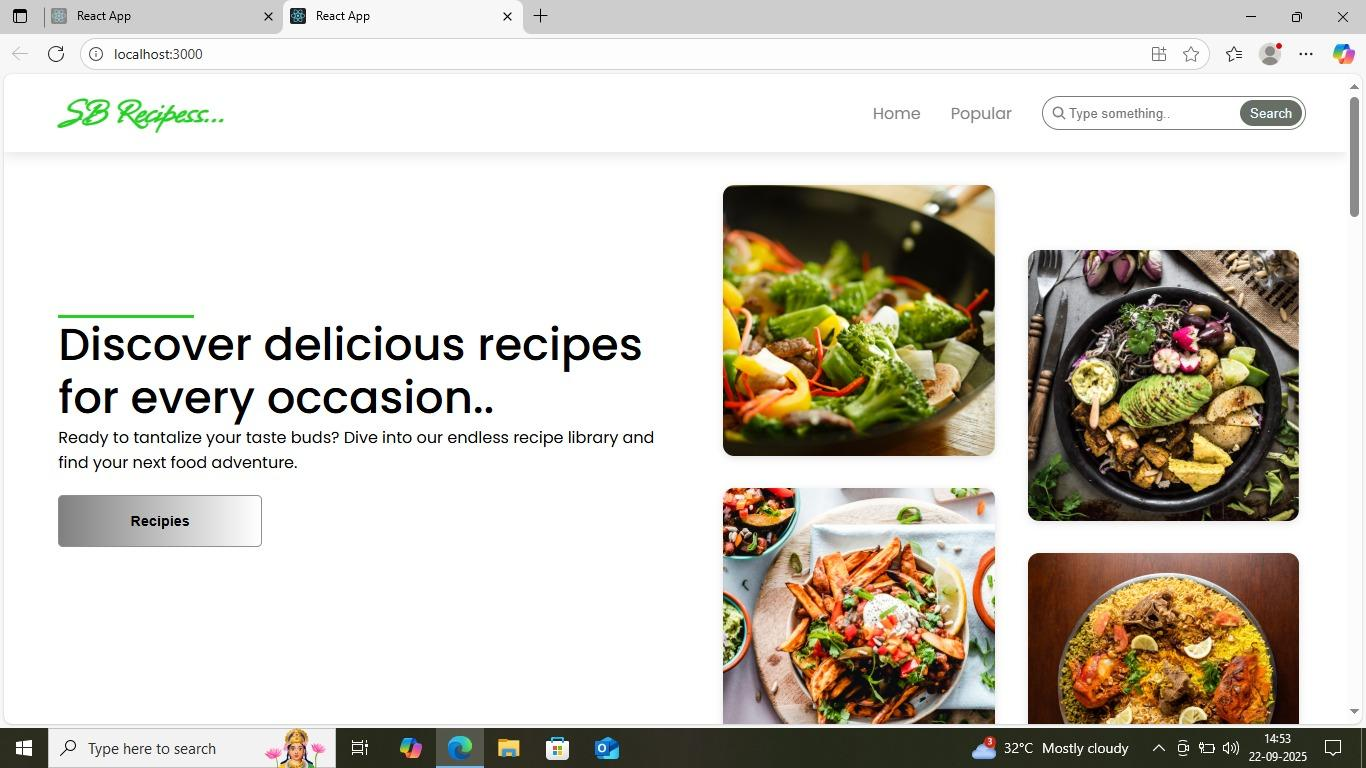
Home Page – Featured recipes and categories.

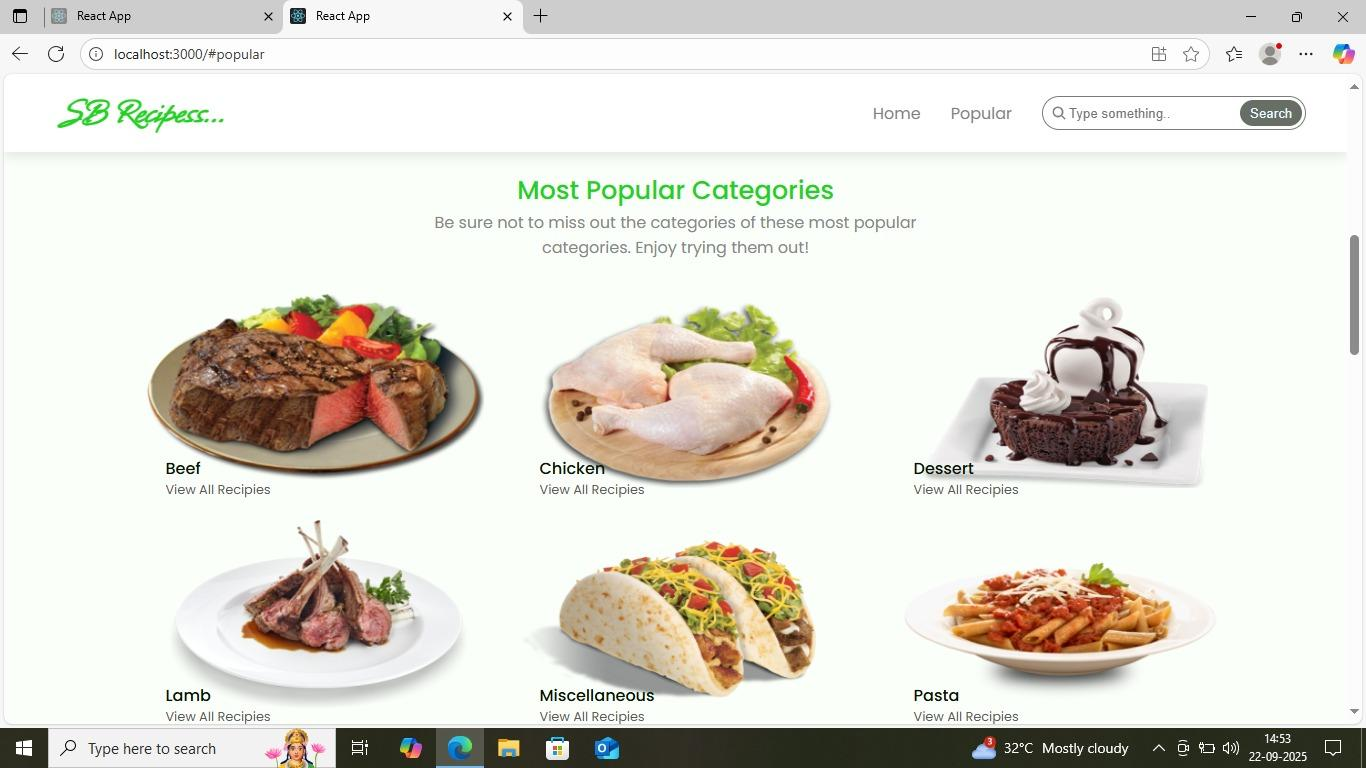
Recipe Search – Search by ingredient, cuisine, or meal type.

Recipe Details Page – Step-by-step instructions, ingredients list, nutrition facts.

User Dashboard – Saved recipes, meal planner, and grocery list generator.

Screenshots / GIFs





Homepage UI with recipe cards.

Search results with filters.

Interactive cooking assistant (step-by-step guide).

10. Styling

CSS Frameworks/Libraries

Tailwind CSS for utility-first responsive design.

Custom Components styled with Styled-Components for dynamic theming.

Theming

Supports light and dark mode.

Custom color palette reflecting a “kitchen vibe” (warm tones like orange, green, and beige).

Future-ready for user-defined themes.

11.Testing

Testing Strategy

Unit Tests: Using Jest + React Testing Library to test components (e.g., RecipeCard, SearchBar).

Integration Tests: Validate interactions (e.g., search input updates results).

End-to-End Tests: Cypress used to simulate real-world flows (searching, saving recipes, following instructions).

Code Coverage

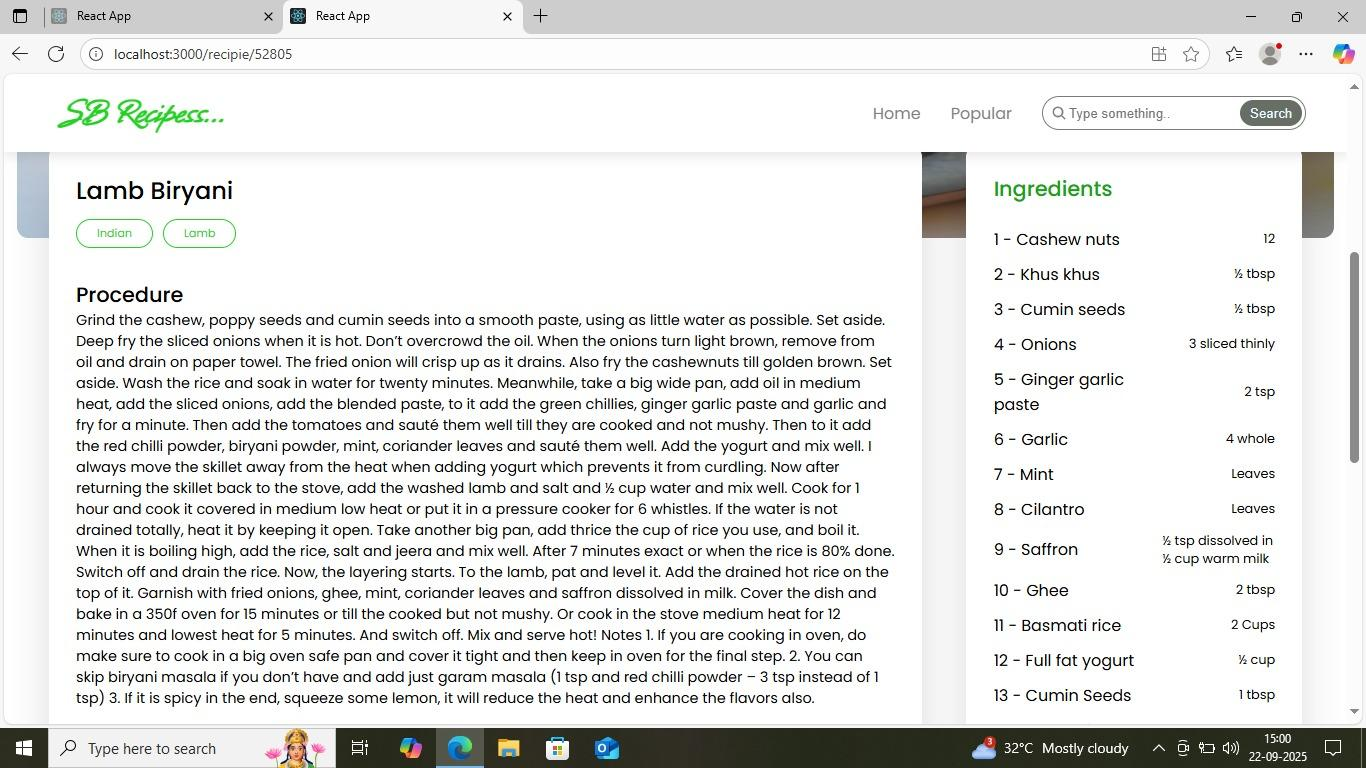
Coverage reports generated with Jest and Istanbul.

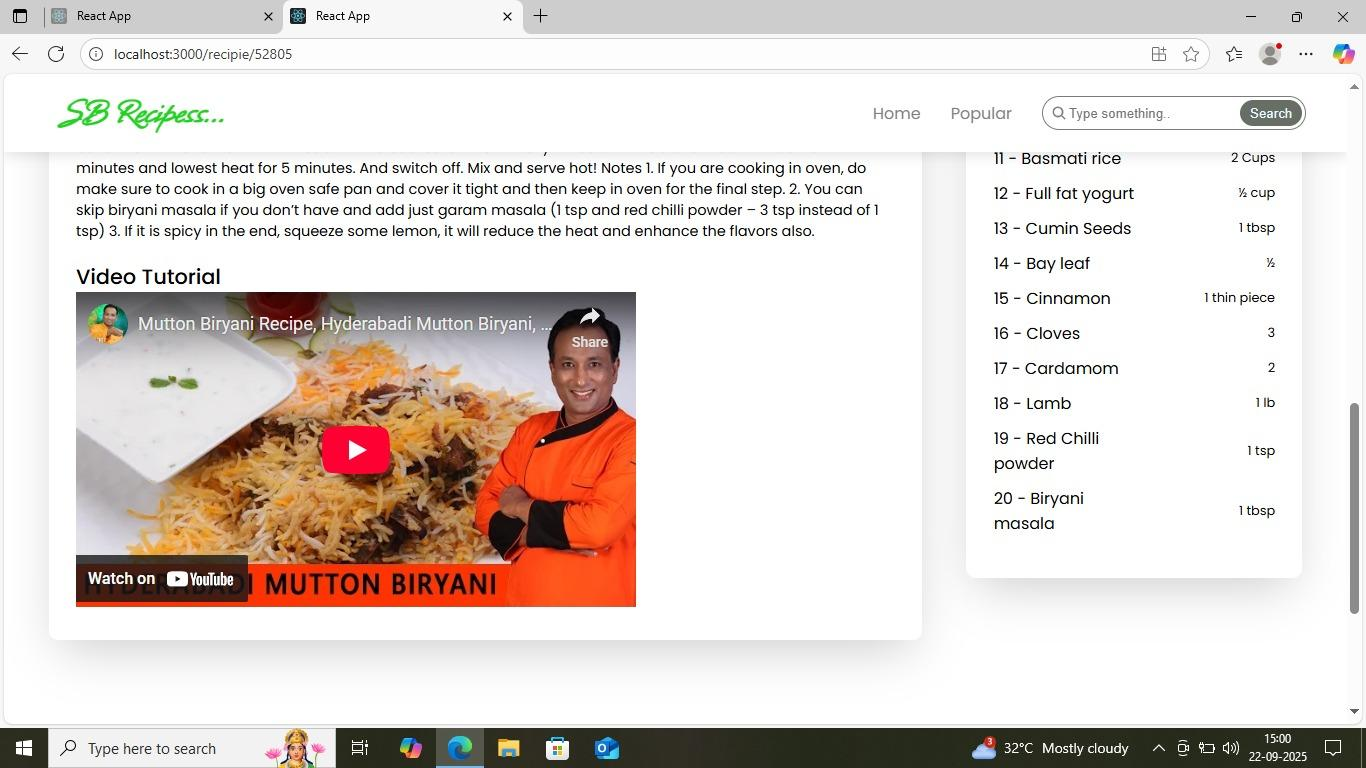
Target: 80%+ coverage across components.

12.Screenshots or Demo

Cookbook – Your Virtual Kitchen Assistant demo link: https://drive.google.com/file/d/1pflnJpRmD\_EG46qCifpMNl6eHCn9aKeJ/view?usp=drivesdk

Screenshots:





13.Known Issues

Dark mode toggle does not persist across sessions.

Some recipe images take longer to load on slower networks.

Grocery list export feature is experimental and may produce duplicates.

14.Future Enhancements

AI-powered meal recommendation engine based on nutrition and preferences.

Voice-guided cooking assistant for hands-free recipe navigation.

Offline support (PWA) for accessing saved recipes without internet.

AR integration for ingredient recognition via camera.

Enhanced animations and transitions using Framer Motion.