**Project Documentation**

1. **Introduction**
   * **Project Title**: Cook Book: Your Virtual Kitchen Assistant
   * **Team ID**: NM2025TMID47871
   * **Team Leader**: AJAYSRI AP ( ajayanimeboy@gmail.com )
   * **Team Members**:
     + 1. DHARANITHARAN R ( dharanitharan1107@gmail.com )
       2. KARTHICK M ( karthickkarthi7776@gmail.com )
       3. KABILAN V ( kabikabi2602@gmail.com )

Roles and Responsibilities :

Team Leader – AJAYSRI AP

Oversees project progress and task allocation.

Coordinates team communication and final integration.

Ensures timely submission and quality of deliverables.

Team Member – DHARANITHARAN R

Designs and develops UI/UX.

Implements music dashboard and playlist features.

Integrates frontend with backend APIs.

Team Member – KARTHICK M

Develops server-side logic and database.

Creates APIs for user, music, and playlists.

Manages authentication and data flow.

Team Member – KABILAN V

Conducts testing and debugging.

Ensures application quality and performance.

Prepares research inputs and documentation.

1. **Project Overview**
   * **Purpose**:
     + **Cook Book: Your Virtual Kitchen** is an interactive culinary companion application designed to help users discover, organize, and prepare meals effortlessly. It aims to elevate the cooking experience by offering curated recipes, personalized meal suggestions, and smart kitchen management features.
   * **Features**:
     + Recipe browsing and cooking guidance
     + Personalized meal recommendations
     + Smart grocery list generation
     + Pantry tracking and ingredient management
     + Search and filter by cuisine, diet, or prep time
     + User profile and saved recipe management
     + Admin panel for recipe and content moderation
2. **Architecture**
   * **Component Structure**: The application follows a modular React component architecture, promoting reusability and separa tion of concerns. Major components include:
     + Header & Navigation – Provides site-wide navigation and access to user profile/settings
     + Recipe List – Displays a grid or list of available recipes with filtering options
     + Recipe Detail – Shows detailed information about a selected recipe, including ingredients and instructions
     + User Profile – Manages saved recipes, preferences, and dietary settings
     + Admin Panel – Restricted component for managing recipe content and user submissions
   * **State Management**: The application uses Context API for global state management. Key state domains include:
     + User Authentication & Profile Data
     + Recipe Collection & Favorites
     + Pantry Inventory
     + Meal Planning Schedule

Context Providers wrap the main application tree, allowing child components to access and update shared state without prop drilling. Local component state is used for UI-specific interactions (e.g., modal visibility, form inputs).

* + **Routing**: Routing is the process of defining how your server responds to different HTTP requests (like GET, POST, etc.) at specific URLs.
    - Using Express.js :

Express is a popular framework built on Node.js that simplifies routing:

Ex :

const express = require('express');

const app = express();

app.get('/home', (req, res) => {

res.send('Welcome to Home Page');

});

app.listen(3000, () => {

console.log('Server running on port 3000');

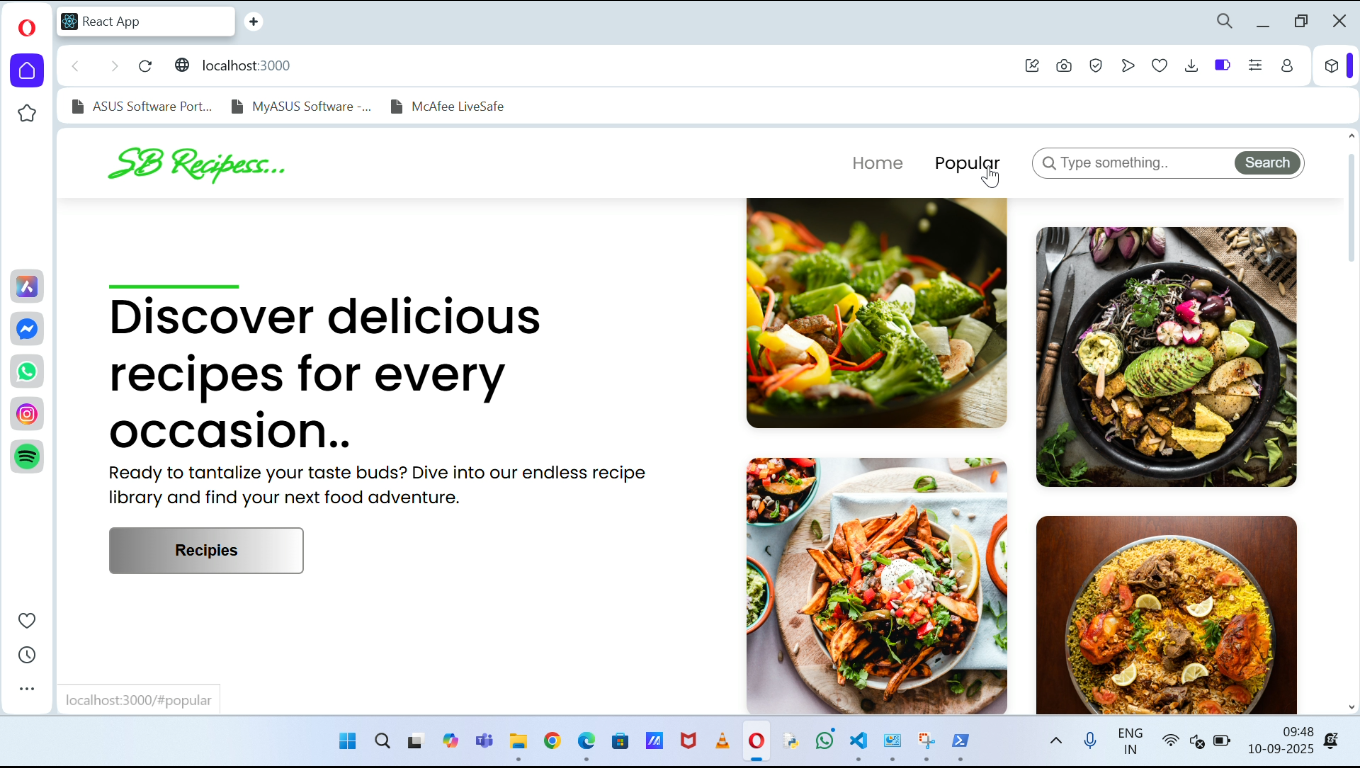
});

1. **Setup Instructions**
   * **Prerequisites**: Node.js, MongoDB, Git, React.js, Express.js, Visual Studio Code
   * **Installation**:

# Clone the repository

# Install client dependencies → cd client && npm install

# Install server dependencies → cd ../server && npm install

1. **Folder Structure**
   * **Client**: The React app is organized into clear folders:
     + - components/ – Reusable parts of the UI like buttons, headers, and recipe cards
       - pages/ – Full screens like Home, Recipes, Pantry, Planner, and Profile
       - assets/ – Images, icons, and other static files
       - styles/ – CSS files or styled-components for design
       - routes/ – Handles navigation between pages
       - context/ – Manages shared data like user info and pantry items
   * **Utilities**: Helpful code is stored in:
     + - utils/ – Functions for filtering recipes, formatting ingredients, etc.
       - hooks/ – Custom React hooks like usePantrySync or useAuth
       - services/ – API calls like fetching recipes or updating user data
2. **Running the Application**
   * Frontend: cd client && npm start
   * Backend: cd server && npm start
   * Access: Visit http://localhost:3000
3. **Component Documentation**
   * **Key Components**:
     + Recipe List –
       - * Displays all available recipes.
         * Props: recipes, on Select Recipe
     + Recipe Detail –
       - * Shows full details of a selected recipe.
         * Props: recipe Id
     + Pantry Manager –
       - * Let users view and update their pantry items.
         * Props: pantry Items, on Update Item
     + Meal Planner –
       - * Helps users plan meals and generate grocery lists.
         * Props: planned Meals, on Add Meal
     + User Profile –
       - * Manages user settings and saved recipes.
         * Props: user Data, on Update Profile
   * **Reusable Components**:
     + Header – Top navigation bar with links and user info. Config: Accepts title, userName, and navigation links.
     + RecipeCard – Compact recipe preview used in lists. Props: title, image, onClick
     + Button – Custom button with styling options. Props: label, onClick, variant
     + Modal – Popup for forms or messages. Props: isOpen, onClose, children
4. **Authentication**
   * JWT-based authentication for secure login. Middleware to protect user and admin routes.
5. **User Interface**
   * Home Page – Displays featured recipes and navigation options
   * Recipe Detail Page – Shows ingredients, instructions, and cooking time
   * Popular – Let users to see popular dishes
   * Meal Planner – Allows users to schedule meals and generate grocery lists
   * Forms – Includes login, recipe submission, and profile update forms
6. **Testing**
   * Manual Testing - Manual testing was performed at key development milestones to ensure core features like recipe browsing, pantry updates, and meal planning worked as expected.
   * Tools Used –
     + Postman – Used to test and verify API endpoints for recipe data, user authentication, and pantry management.
     + Chrome Dev Tools – Used for debugging UI components, inspecting network requests, and monitoring performance
7. **Screenshots or Demo**
8. **Known Issues**

* Pantry sync delay - Updates to pantry items may take a few seconds to reflect across all components.
* Mobile responsiveness - Some UI elements may not display correctly on smaller screens and require layout adjustments.

1. **Future Enhancements**

* New Components: Add features like a recipe rating system, comment section, and step-by-step cooking timers
* Enhanced Styling: Improve visual design with animations, transitions, and theme customization option
* Mobile Optimization: Refine layout and responsiveness for better performance on smartphones and tablets
* Notifications: Introduce reminders for meal planning, pantry restocks, and recipe suggestions