COOKBOOK : YOUR VIRTUAL KITCHEN ASSISTANT

INTRODUCTION:

Project Title: cookbook - your virtual kitchen assistant

Team ID: NM2025TMID229921

Team Leader: JANANI S - 202400702@sigc.edu

Team members:

KANIMOZHI K - 202400176@sigc.edu

KAVIYA P-202400910@sigc.edu

KAVIPRIYA P-202400040@sigc.edu

2. PROJECT VIEW:

Purpose:

The Cookbook App aims to provide users with a comprehensive and interactive platform for discovering, organizing, and sharing recipes. Whether you're a beginner in the kitchen or an experienced cook, this app helps users find new recipes based on their preferences, dietary needs, and available ingredients. It offers personalized recommendations, allows users to save and manage their favorite recipes, and even creates shopping lists for ingredients.

Features:

The Cookbook App frontend comes with several key features to enhance the user experience

tried, reviewed, or Recipe Discovery and Search:

Advanced Filters: Users can filter recipes by dietary restrictions (e.g., vegetarian, gluten-free), difficulty level, cooking time, and ratings.

Recipe Categories: Predefined categories like "Appetizers," "Main Courses," "Desserts," etc., for quick browsing.

Recipe Details:

Ingredients List: Shows the full list of ingredients, with quantities, needed for the recipes.

Nutritional Information: Displays key nutrition details (calories, protein, carbs, etc.) for health-conscious users.

Cooking History:

A log of recipes the user has saved.

Custom Preferences: Users can set dietary restrictions, preferred cuisines, or favorite ingredients.

Shopping List Integration:

Add Ingredients to Shopping List: Users can add recipe ingredients to a shopping list with one click.

Shopping List Management: Allows users to check off items as they shop or modify the list.

Recipe Sharing:

Users can share recipes with others, either within the app or via social media.

User-Contributed Recipes: If the app allows community submissions, users can share their own recipes, including pictures, ingredients, and instructions.

3. ARCHITECTURE:

Component Structure:

Navbar → Handles site navigation

HomePage → Displays featured and trending recipes

RecipeList → Shows list of recipes fetched from API

RecipeDetail → Provides detailed view of selected recipe

Favorites → Stores bookmarked recipes

ShoppingList → Manages ingredients for shopping.

State Management:

Context API for global state (favorites, shopping list)

useState and useReducer for local state within components

Routing (React Router v6):

/ → Home

/recipes → Recipe List

/recipe/:id → Recipe Detail

/favorites → Favorites Page

/shopping-list → Shopping List

4. SETUP INSTRUCTIONS:

Prerequisites:

Before setting up the Cookbook App locally, ensure that you have the following software installed on your system:

Node.js and npm:

Node.js is a JavaScript runtime required to run the app. npm (Node Package Manager) comes bundled with Node.js to manage dependencies.

To check if Node.js is installed, run the following command:

node -v

To check if npm is installed, run:

npm -v

Installation:

Follow these steps to set up the Cookbook App locally:

Clone the Repository:

configure them locally.

Start by cloning the project's repository from GitHub to your local machine.

Replace your-username with the actual GitHub username or organization.

5. FOLDER STRUCTURE:

Code

```
src/
 ├── assets/          # Static files like images, fonts, icons
 |   └── images/
```

```
├── components/      # Reusable UI components (e.g., Button, Modal)
|   └── RecipeCard/
├── pages/           # Route-level components (e.g., Home, RecipeDetail)
|   └── Home/
|   └── RecipeDetail/
├── hooks/           # Custom React hooks
|   └── useFetchRecipes.js
├── utils/           # Helper functions and constants
|   └── formatDate.js
|   └── constants.js
├── services/        # API calls and external integrations
|   └── recipeService.js
├── context/         # React Contexts for global state
|   └── RecipeContext.js
├── styles/          # Global and shared styles
|   └── variables.css
├── App.js           # Root component
├── index.js         # Entry point
└── routes.js        # Route definitions
```

Utilities:

These are the behind-the-scenes heroes that make your app efficient and DRY (Don't Repeat Yourself):

public/: Typically holds the static files, including index.html, favicon, and other assets that are referenced directly.

src/: The main source directory containing all the React-related code.

assets/: Images, icons, and styles (like CSS, SCSS, or styled-components) go here.

components/: Reusable UI components like buttons, headers, cards, etc. These should ideally be small, independent, and reusable.

pages/: This folder contains full page-level components. These are generally composed of smaller components from the components/ directory and are used to represent entire pages.

context/: Houses all your context providers if you're using React Context API for global state management.

hooks/: Custom hooks like useFetch, useForm, useLocalStorage, etc., for reusable logic.

utils/: Utility functions like formatting helpers, validation functions, and other helpers that are not necessarily tied to a specific component.

App.js: The main React component that renders your app.

index.js: The entry point of the app, where ReactDOM renders the root component into the DOM.

package.json: Manages your project dependencies and scripts.

SAMPLE RECIPES:

Chicken enchilada casserole

Procedure:

Cut each chicken breast in about 3 pieces, so that it cooks faster and put it in a small pot. Pour Enchilada sauce over it and cook covered on low to medium heat until chicken is cooked through, about 20 minutes. No water is needed, the chicken will cook in the Enchilada sauce. Make sure you stir occasionally so that it doesn't stick to the bottom. Remove chicken from the pot and shred with two forks. Preheat oven to 375 F degrees. Start layering the casserole. Start with about ¼ cup of the leftover Enchilada sauce over the bottom of a baking dish. I used a longer baking dish, so that I can put 2 corn tortillas across. Place 2 tortillas on the bottom, top with ⅓ of the chicken and ⅓ of the remaining sauce. Sprinkle with ⅓ of the cheese and repeat starting with 2 more tortillas, then chicken, sauce, cheese. Repeat with last layer with the remaining ingredients, tortillas, chicken, sauce and

cheese. Bake for 20 to 30 minutes uncovered, until bubbly and cheese has melted and started to brown on top. Serve warm.

Ingredients:

- Enchilada sauce - 14 oz jar
- shredded Monterey Jack - 3 Cups
- corn tortillas - 6
- chicken breast - 2

Christmas pudding trifle

Procedure:

Peel the oranges using a sharp knife, ensuring all the pith is removed. Slice as thinly as possible and arrange over a dinner plate. Sprinkle with the demerara sugar followed by the Grand Marnier and set aside. Crumble the Christmas pudding into large pieces and scatter over the bottom of a trifle bowl. Lift the oranges onto the pudding in a layer and pour over any juices. Beat the mascarpone until smooth, then stir in the custard. Spoon the mixture over the top of the oranges. Lightly whip the cream and spoon over the custard. Sprinkle with the flaked almonds and grated chocolate. You can make this a day in advance if you like, chill until ready to serve.

Ingredients:

- Orange - 3

- Demerara Sugar - 1 tbs
- Grand Marnie - 2 tbs
- Christmas Pudding - 300g
- Custard - 500g
- Mascarpone - 250g
- Double Cream - 284ml
- Flaked Almonds Handful
- Dark Chocolate Grated

Brie wrappedin peosciutto & brioche

Procedure:

Mix the flour, 1 tsp salt, caster sugar, yeast, milk and eggs together in a mixer using the dough attachment for 5 mins until the dough is smooth. Add the butter and mix for a further 4 mins on medium speed. Scrape the dough bowl and mix again for 1 min. Place the dough in a container, cover with cling film and leave in the fridge for at least 6 hrs before using. Wrap the Brie in the prosciutto and set aside. Turn out the dough onto a lightly floured surface. Roll into a 25cm circle. Place the wrapped Brie in the middle of the circle and fold the edges in neatly. Put the parcel onto a baking tray lined with baking parchment and brush with beaten egg. Chill in the fridge for 30 mins, then brush again with beaten egg and chill for a further 30 mins. Leave to rise for 1 hr at room temperature. Heat oven to 200C/180C fan/gas 6, then bake for 22 mins. Serve warm.

Ingredients:

- Plain Flour - 375g
- Caster Sugar - 50g
- Yeast - 7g
- Milk -75g
- Eggs -3 Large
- Eggs To Glaze
- Butter - 180g
- Brie -250g
- Prosciutto - 8 slices

6. RUNNING THE APPLICATION:

1. Navigate to the Client Directory

First, ensure that you're in the correct project directory. If you've followed the previous setup instructions and installed the necessary dependencies, navigate to the client directory (if applicable).

2. Start the Frontend Server

Once you are in the correct directory (either the root or the client folder), run the following command to start the frontend development server

npm start in the client directory.

7.COMPONENT DOCUMENTATION:

Key Components:

RecipeCard – Displays recipe summary

RecipeDetails – Shows cooking steps

ShoppingList – Generates list from selected recipes

UserProfile – Stores user preferences

Reusable Components:

Buttons, Modals, SearchBar, Filters

8. STATE MANGEMENT :

Global State: State management is a key concept in modern web development, especially when building React applications or working with frameworks that involve dynamic data flow.

Let's break down **Global State** and **Local State** in more detail.

Global state refers to the state that is accessible throughout the entire application or across multiple components. This type of state is used when you want to share data between many parts of the application.

How Global State Works:

Global state is often managed using a **state management library** or built-in solutions in a framework, such as **React Context**, **Redux**, **Zustand**, or **Recoil**.

Local State:

Local state refers to the state that is specific to a single component. It is used for managing UI-related or component-specific data that doesn't need to be shared across the entire app. Local state is often stored inside individual components and is independent of other components.

How Local State Works:

- Local state is usually managed using hooks like useState in React or internal state mechanisms provided by the framework.

9. USER INTERFACE(UI):

1. Form UI

Forms are essential elements in most apps for capturing user input. Below is a breakdown of common form features that can be showcased in a UI:

Features:

- **Text Fields** (e.g., name, email)
- **Password Field** (with eye icon to toggle visibility)

2. Pages (Layouts)

Pages refer to the structural layout of the app or website. A good UI needs intuitive and clear layouts for different functionalities.

Features:

- **Navigation Bars** (top or side)
- **Content Section** (header, body, footer)

3. Interactions

UI interactions provide feedback to users based on their actions (clicking, hovering, submitting). These interactions can include modal windows, tooltips, sliders, or buttons with hover effects.

Features:

- **Hover Effects** (e.g., buttons changing color when hovered)
- **Modals/Pop-ups** (to display additional information or actions)

10. STYLING:

In the Cookbook App, styling is essential to ensure a clean, responsive, and intuitive user interface. Here's an overview of the CSS frameworks, libraries, and theming techniques used:

CSS Frameworks/Libraries

CSS Modules (Default Styling Approach):

Theming:

Custom Design System:The app follows a custom design system to ensure consistency in visual elements like colors, typography, spacing, and component behavior across the app.

Design System Elements:Colors: A predefined color palette with primary, secondary, and accent colors.

11. TESTING STRATEGY :

Testing is a crucial part of the development process to ensure that applications work as expected, are robust, and are free of bugsHere's how you can approach testing in a modern React application, including **unit**, **integration**, and **end-to-end** testing.

Testing Strategy Overview:

There are three main levels of testing:

1. **Unit Testing**
2. **Integration Testing**
3. **End-to-End (E2E) Testing**

1. Unit Testing

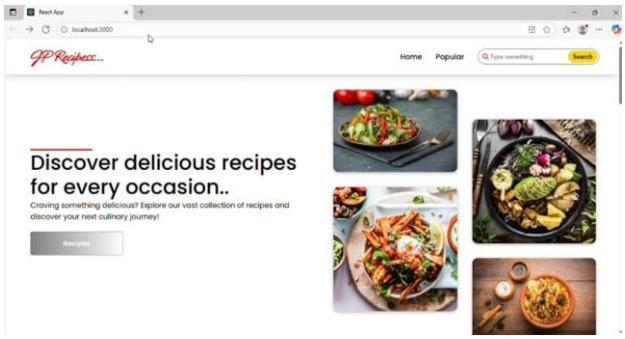
Unit testing focuses on testing individual components or functions in isolation. The goal is to verify that each part of the application works correctly by itself, before integrating it with other parts.

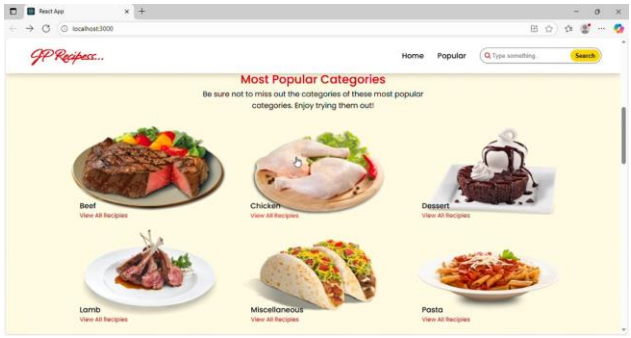
Tools Used:

- **Jest**: A JavaScript testing framework, often used for unit testing in React applications.

React Testing Library: Provides utilities to interact with and test React components in a way that simulates real user interactions.

12. SCREENSHOTS &DEMO:





Demo Link

https://drive.google.com/file/d/1KYbHno6r974nFpBAtwn06O0qCPuYpg24/view?usp=drivesdk

13.KNOWN ISSUES:

1. API Rate Limits

Issue: Spoonacular API has strict rate limits for free plans.

Impact: Frequent requests (e.g., live search or filtering) may result in failed fetches or 429 errors.

Workaround: Implement request throttling or caching.

2. Image Upload Failures

Issue: Uploading large images or unsupported formats may fail silently.

Impact: Recipe submission may appear successful but lack images.

Workaround: Add client-side validation and feedback for file size/type.

3. Search Debounce Lag

Issue: SearchBar with debounce may delay updates or feel unresponsive.

Impact: Poor UX during rapid typing.Workaround: Tune debounce timing and provide loading indicators.

14.FUTURE ENHANCEMENTS:

1. New Features & Functionalities

User Authentication & Authorization:

Sign Up / Login: Implement user authentication so users can sign up, log in, and access their personal data (e.g., saved recipes, preferences, shopping lists).

OAuth Integration: Allow users to log in via third-party services (e.g., Google, Facebook).

Password Reset & Profile Management: Allow users to manage their accounts, update preferences, and reset passwords.

Advanced Recipe Search & Filters:

Ingredient-Based Search: Allow users to search for recipes based on the ingredients they have at home (e.g., "Recipes with chicken and tomatoes").

Dietary Preferences: Add more granular search filters for specific diets like keto, vegan, gluten-free, etc.

Seasonal Recipes: Display recipes based on the current season, utilizing seasonal ingredients

Nutrition Information:

Integrate with an API (e.g., Edamam, Spoonacular) to display nutrition information (calories, macros, vitamins) for each recipe.

Allow users to track their daily or weekly nutrition intake based on the recipes they've prepared.

Social Sharing & Community Features:

Recipe Sharing: Let users share their favorite recipes with friends and family via social media or a unique link.

Recipe Comments & Reviews: Allow users to leave comments, ratings, and reviews on recipes they have tried.

Recipe Collections: Users can create and share their own recipe collections (e.g., “Quick Dinner Ideas,” “Holiday Baking”).

Cooking Timer & Voice Assistance:

Built-in Timer: Add a built-in timer that users can use to time each step of the cooking process.

Voice Assistance: Integrate voice commands or integration with voice assistants (e.g., Alexa, Google Assistant) to help users through the recipe instructions hands-free.

15.CONCLUSION:

The CookBook: Your Virtual Kitchen Assistant project showcases the strength of React.js in building an engaging and scalable frontend application. With its clean component structure, efficient state management using Context API, and responsive UI, it provides an enjoyable user experience for recipe exploration and cooking guidance.

While the current version offers essential features such as recipe browsing, favorites, and shopping list management, the planned future enhancements will further improve personalization, accessibility, and usability. This project demonstrates how modern frontend technologies can simplify everyday activities like cooking, making it more interactive and fun for users worldwide.