Frontend Development with React.js Project Documentation

**Introduction**

**Project Title**: Cook Book: Your Virtual Kitchen Assistant.

TEAM ID: NM2025TMID42166

TEAM LEADER: PRIYADHARSHINI D

ROLE: CODING AND DEVELOPMENT

TEAM MEMBER: RENUGA L

ROLE: UPLOADING

TEAM MEMBER: SALMABANU Z

ROLE: DEMO VIDEO

TEAM MEMBER: SATHIYA P

ROLE: DOCUMENT CREATER

Project Overview

Purpose: The purpose of the Cook Book Virtual Assistant is to provide users with an easy-to-use application that helps them explore, organize, and prepare recipes with step-by-step guidance.

Features: Recipe search, Meal planning, Shopping list, Voice guidance and personalization.

Architecture

Component Structure: Recipe browsing and search

Meal planning and shopping list creation

Step-by-step cooking assistant with voice guidance

State Management: The application uses the Context API (or Redux, depending on scale) for global state management to handle

Routing: React Router is used to manage navigation between different views, such as:

Shop list

Setup Instructions

Prerequisites: The required software dependencies are Node.js and np

,React.js.

Installation: Provide a step-by-step guide to Clone the repository from GitHub. Navigate to the client directory.

\* Run npm install to install dependencies.

\* Configure environment variables for any API keys

Folder Structure

Client: The main React application is organized into folders such as components, pages, and assets for images and audio files.

Utilities: Helper functions for music playback and custom hooks for managing audio state are located in the utils folder

Code:

project-name/

├── public/

│ ├── index.html

│ └── favicon.ico

├── src/

│ ├── assets/

│ │ ├── images/

│ │ └── styles/

│ ├── components/

│ │ ├── Header.js

│ │ ├── Footer.js

│ │ └── ...

│ ├── containers/

│ │ ├── App.js

│ │ └── ...

│ ├── actions/

│ │ ├── index.js

│ │ └── ...

│ ├── reducers/

│ │ ├── index.js

│ │ └── ...

│ ├── utils/

│ │ ├── api.js

│ │ └── ...

│ ├── index.js

│ └── setupTests.js

├── tests/

│ ├── components/

│ │ ├── Header.test.js

│ │ └── ...

│ └── ...

├── .git.ignore

├── package. json

└── README.md

Running the Application

Provide commands to start the frontend server locally.

Frontend: cd client

npm start

Component Documentation

Key Components: Recipe Viewer Component: Displays recipe details (ingredients, instructions, cooking time).

State Management

Global State: ..Global State:

The Context API (or Redux) is used to manage the global state, ensuring that:

Local State: Components use use State hooks to manage their own local state, such as form inputs (e.g., adding a new recipe) or UI toggles (e.g., expanding ingredient steps).

User Interface

Screenshots or GIFs can be included to showcase:

Home Page: Featured and recommended recipes.

Recipe Detail Page: Ingredients, cooking instructions, and nutrition info.

Shopping List Page: Auto-generated grocery list.

Meal Planner Page: Weekly meal schedule view.

Styling

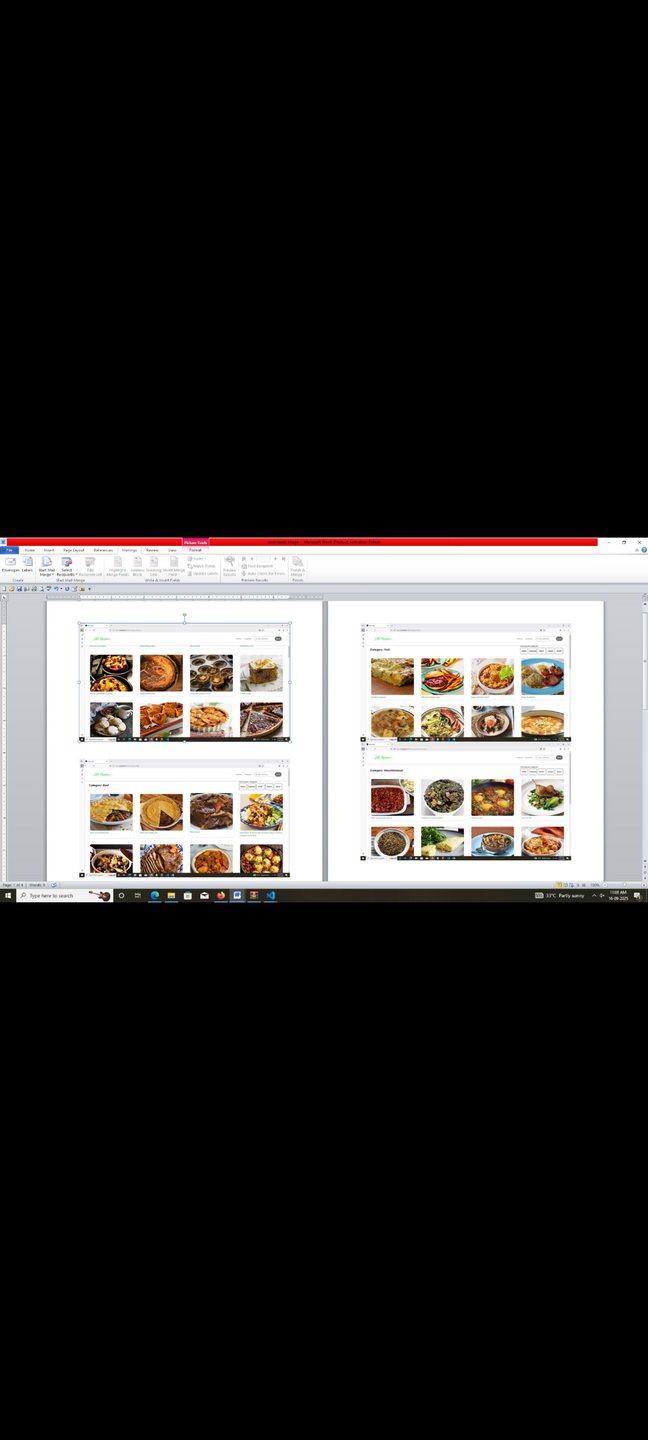
CSS Frameworks/Libraries: Describe any CSS frameworks, libraries, or pre- processors (e.g., Sass, Styled-Components) used.

Theming: A custom design system with light and dark themes is implemented

Testing

Testing Strategy: The application uses Jest and React Testing Library for unit and integration testing of components.

Code Coverage: Code coverage is tracked using Istanbul/nyc to ensure all key components are adequately tested.

Screenshots or Demo: 

Favorites:



1. Known Issues
   * Any known bugs or issues, such as occasional playback glitches on certain browsers, should be documented here
   * Future Enhancements
   * User Authentication System – Allow users to sign in, save recipes, and sync data across devices.

Offline Mode – Save recipes locally for offline cooking assistance.

1. DEMOLINK:  https://drive.google.com/file/d/1wjXooXeMVWuVWJZPnIsTLUObU3fekCAc/view?usp=sharing