# Cookbook Recipe Website - Project Documentation

## Introduction

Project Title: Cookbook Recipe Website

Team Members:  
Varsha K -- (Project Lead, JS Developer)  
Vedha Shree R -- (HTML Developer)  
Shree Pooja S -- (UI/UX Designer)  
Keerthna C -- (CSS Developer)  
Soumya S -- (Testing and Quality Assurance)

## Project Overview

### Purpose

The purpose of this project is to design a user-friendly recipe website that showcases various dishes across categories such as chicken, beef, lamb, pasta, soups, and more. The system allows users to browse recipes, view detailed ingredients and instructions, and watch video tutorials.

### Features

- Home Page (cookbook.html) with hero banner and navigation.  
- Categories section for different recipe types (Beef, Chicken, Pasta, etc.).  
- Individual recipe detail pages (e.g., chicken curry, lamb biryani, mutton soup).  
- Embedded YouTube video tutorials for step-by-step cooking guidance.  
- Search functionality for recipes.  
- Custom CSS styling for improved design and responsiveness.

## Architecture

### Component Structure

The website consists of multiple HTML pages linked together:  
- cookbook.html – Acts as the homepage, displaying categories and trending dishes.  
- beef.html, chicken.html – Display grouped recipes by category.  
- Individual recipe pages (e.g., chickencurry.html, lambbriyani.html, Spaghetti-Meatballs.html).  
- style.css – Provides styling for layout, responsiveness, and UI components.

### State Management

Currently, the website is static and does not use any advanced state management. Future versions may integrate JavaScript or a frontend framework like React for dynamic state handling.

### Routing

Routing is achieved using standard HTML links between pages. Future enhancements may include React Router for smoother navigation.

## Setup Instructions

### Prerequisites

- A modern web browser (Chrome, Firefox, Edge)  
- Visual Studio Code (VS Code)  
- VS Code Live Server extension (optional)

### Installation

1. Download the project folder.  
2. Ensure all HTML, CSS, and assets are in the same directory.  
3. Open cookbook.html in the browser to start.  
4. Alternatively, open the project in VS Code and run with Live Server.

## Folder Structure

Project Folder:  
- cookbook.html (Home Page)  
- beef.html, chicken.html (Category Pages)  
- lambbriyani.html, muttonsoup.html, Spaghetti-Meatballs.html, VegSoup.html (Recipe Pages)  
- chicken-recipe/ (Contains multiple chicken recipes like curry, biryani, fried rice, etc.)  
- style.css (Global styling)  
- assets/ (Images, icons, etc.)

## Running the Application

Option 1 – Run via Live Server in VS Code (Recommended):  
Open the project folder in VS Code. Right-click on cookbook.html and select 'Open with Live Server'.  
  
Option 2 – Run directly in a Web Browser:  
Double-click cookbook.html to open in your browser and navigate through links.

## Component Documentation

cookbook.html – Provides navigation, hero section, category grid, and trending dishes.  
  
beef.html – Displays beef-related recipes with images and links.  
  
chicken.html – Displays chicken-related recipes with cards linking to details.  
  
Recipe detail pages – Each includes ingredients, instructions, tags, and embedded video tutorial.  
  
style.css – Centralized stylesheet handling design consistency.

## Styling

The project uses a custom CSS file (style.css) to style the navigation bar, hero section, categories, recipe cards, and footer. The layout is responsive with grid-based sections and hover animations. Future enhancements may use frameworks like Tailwind CSS or Bootstrap.

## Testing

Manual testing has been conducted:  
- Navigation between pages works correctly.  
- Recipe cards link to correct detail pages.  
- Video tutorials load and play as expected.  
- Layout and responsiveness verified on different screen sizes.

## Screenshots or Demo

Screenshots of the homepage, category pages, and recipe details can be included here.

## Known Issues

- Static design with no backend support.  
- No user authentication or personalization.  
- Recipes are hardcoded and not dynamically fetched.  
- Limited search functionality.

## Future Enhancements

- Convert the project to a React.js frontend with reusable components.  
- Add global state management (e.g., Redux, Context API).  
- Integrate a backend API for dynamic recipe loading.  
- Implement user login, favorites, and reviews.  
- Enhance UI with modern frameworks and animations.