# Cookbook Project Documentation

### Introduction

**Project Title:** CookBook: Your Virtual Kitchen Assistant

**Team Members:**

**Team ID :** 149958

|  |  |
| --- | --- |
| Team Members | Email Id |
| **Team Leader :** JEEVA N | j6385703768@gmail.com |
| **Team member :** ARSHAD KHAN A | arshadkhan2004official@gmail.com |
| **Team member :** MOHANRAJ M | mohanrajmurugan31@gmail.com |
| **Team member :** HARIHARAN P | itzmemsd07@gmail.com |
| **Team member :** LOGESH P | logeshsharma1155@gmail.com |

**Purpose**

# Project Overview

The **Cookbook** project is a web application built using **React.js with Vite**. It allows users to browse, manage, and share recipes easily. The goal is to provide a user-friendly platform for food lovers and home chefs to organize their favorite recipes.

#### Features

* **User Authentication** - Secure login and signup functionality.
* **Recipe Management** - Users can add, edit, and delete recipes.
* **Search & Filters** - Advanced search options to find recipes easily.
* **Favorites & Bookmarks** - Users can save their favorite recipes.
* **Responsive Design** - Optimized for mobile and desktop use.

**Architecture**

#### Component Structure

The application follows a modular component-based architecture:

* **App.jsx** - The main entry component.
* **Header.jsx** - Navigation bar component.
* **Home.jsx** - Displays a list of recipes.
* **RecipeCard.jsx** - A reusable component for individual recipe display.
* **RecipeForm.jsx** - Form to add/edit recipes.
* **Footer.jsx** - Footer section.

#### State Management

* Uses **Context API** for managing global state.
* RecipeContext.jsx handles recipe-related data across components.
* Local state is used within individual components for UI interactions.

#### Routing

* Uses react-router-dom for navigation between pages.
* Example routes:
  + / → Home Page
  + /recipe/:id → Recipe Details Page
  + /add-recipe → Add New Recipe Page

**Prerequisites**

## Setup Instructions

* Install **Node.js** (latest stable version recommended)

#### Installation

1. Clone the repository:
2. git clone https://github.com/yourusername/cookbook.git
3. Navigate to the project directory:
4. cd cookbook
5. Install dependencies:
6. npm install
7. Start the development server:
8. npm run dev

## Folder Structure

📦 cookbook

┣ 📂 src

┃ ┣ 📂 components

┃ ┃ ┣ 📜 Header.jsx

┃ ┃ ┣ 📜 Footer.jsx

┃ ┃ ┣ 📜 RecipeCard.jsx

┃ ┃ ┗ 📜 RecipeForm.jsx

┃ ┣ 📂 pages

┃ ┃ ┣ 📜 Home.jsx

┃ ┃ ┗ 📜 RecipeDetails.jsx

┃ ┣ 📂 context

┃ ┃ ┗ 📜 RecipeContext.jsx

┃ ┣ 📜 App.jsx

┃ ┗ 📜 main.jsx

┣ 📂 public

┣ 📜 package.json

┣ 📜 vite.config.js

┗ 📜 README.md

### Running the Application

* Run the frontend locally:
* npm run dev

### Component Documentation

#### Key Components

**RecipeCard.jsx**

import React, { useEffect } from 'react'; import YouTube from 'react-youtube'; import axios from 'axios';

import { useNavigate, useParams } from 'react-router-dom';

const Recipie = () => { return (

<>

<div className="recipe-page">

{recipie ? (

<div className="recipe-container">

{/\* Recipe Header: Image and Ingredients \*/}

<div className="recipe-header">

<div className="recipe-img">

<img

src={recipie.strMealThumb} alt="food-item" className="recipe-image"

/>

</div>

<div className="ingredients-container">

<h3>Ingredients</h3>

<ul className="ingredients-list">

{Object.entries(recipie).map(([key, value]) => { if (key.startsWith('strIngredient') && value) {

const ingredientNumber = key.slice('strIngredient'.length); const measure =

recipie[`strMeasure${ingredientNumber}`] || ''; return (

<li key={key} className="ingredient">

<h5>{value}</h5>

<p>{measure}</p>

</li>

);

}

return null;

})}

</ul>

</div>

</div>

{/\* Recipe Details: Procedure, Area, Category \*/}

<div className="recipe-details">

<h4>{recipie.strMeal}</h4>

<div className="recipe-specials">

<p>{recipie.strArea && recipie.strArea}</p>

<p>{recipie.strCategory && recipie.strCategory}</p>

</div>

<div className="procedure">

<h5>Procedure</h5>

<p>{recipie.strInstructions}</p>

</div>

</div>

{/\* YouTube Video \*/}

{recipie.strYoutube && (

<div className="youtube-video-container">

<h5>Video Tutorial</h5>

<YouTube

className="youtube-video" videoId={recipie.strYoutube.slice(32)} opts={{

height: '315',

width: '560',

}}

/>

</div>

)}

</div>

) : (

<p className="loader"></p>

)}

</div>

</>

);

export default Recipie;

#### Reusable Components

* **Header.jsx** - Navigation bar
* **Footer.jsx** - Bottom section
* **RecipeCard.jsx** - Displays a single recipe card
* **RecipeForm.jsx** - Handles recipe creation/editing

### State Management

#### Global State

* Managed using **Context API** in RecipeContext.jsx.
* Provides recipes data to all components.

#### Local State

* Used within components for UI handling, such as form inputs.

## User Interface

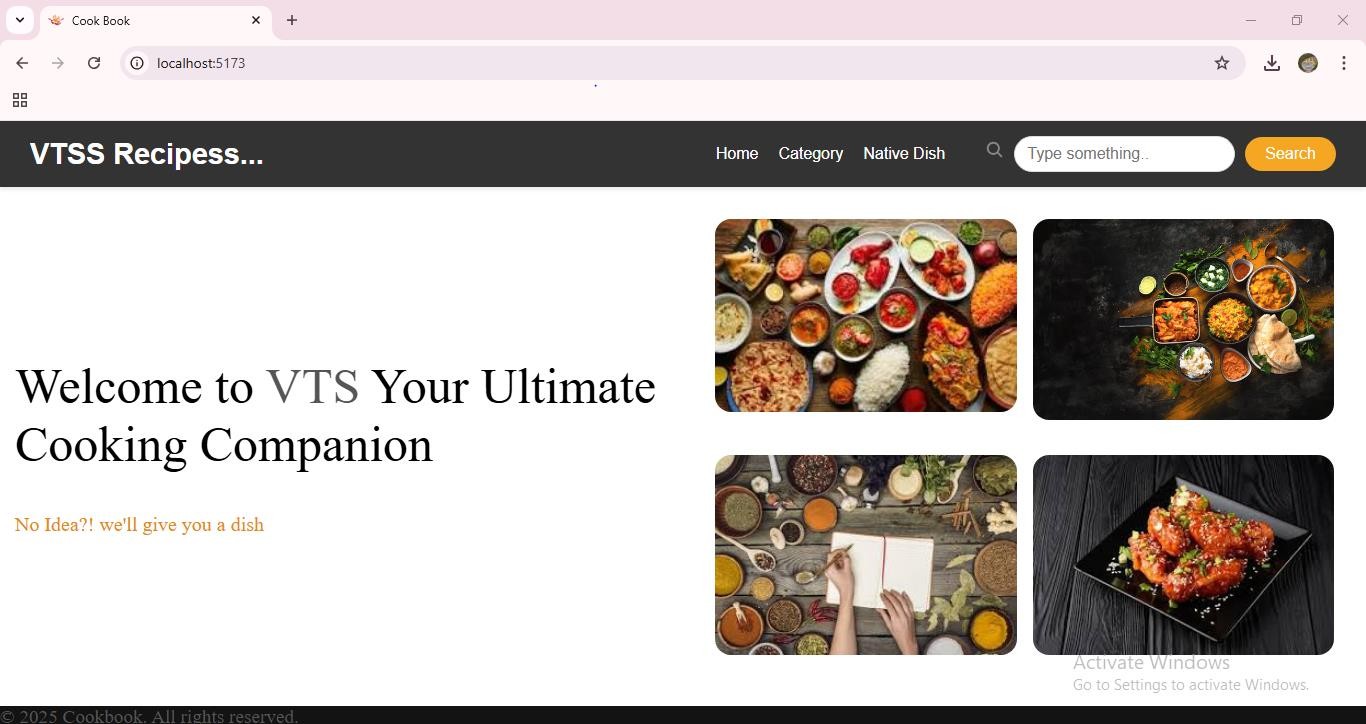
**Styling**

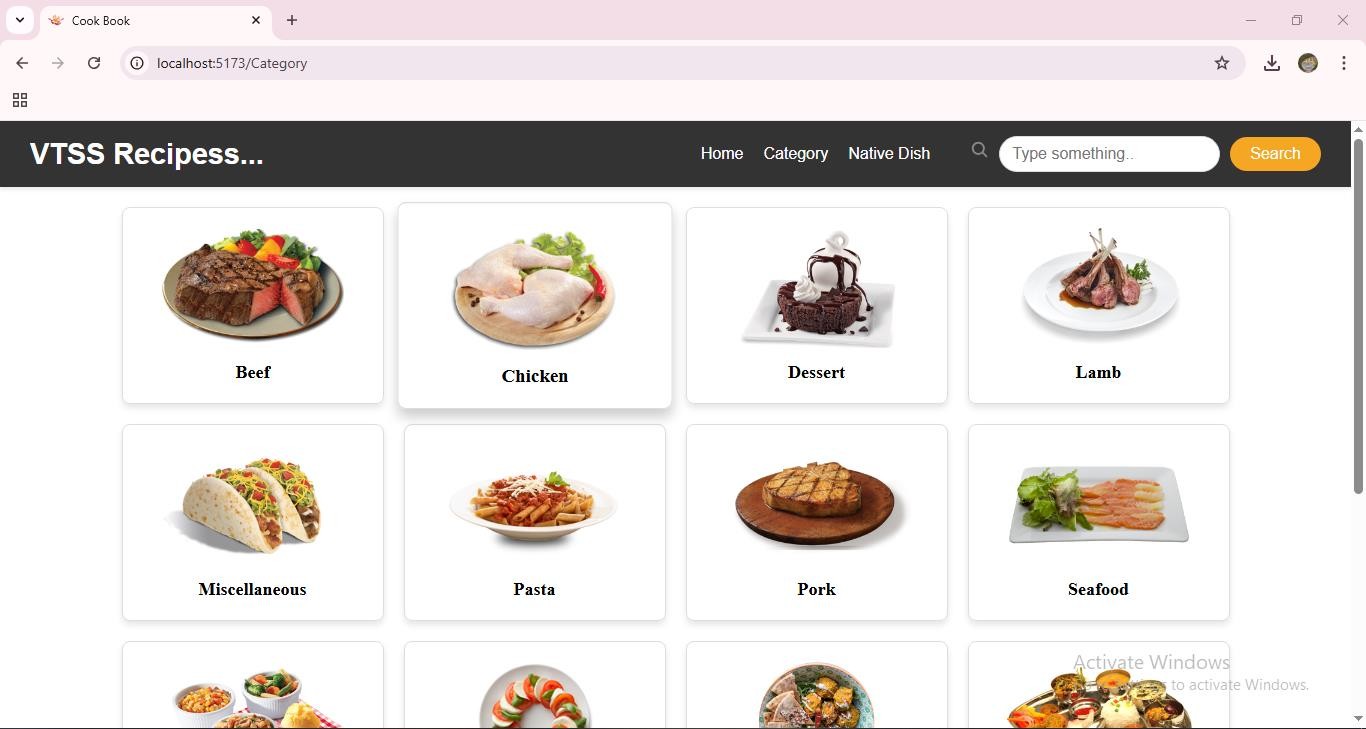
* **CSS: Normal** CSS.
* **Theming:** Custom styles for buttons, cards, and navigation.

**Testing**

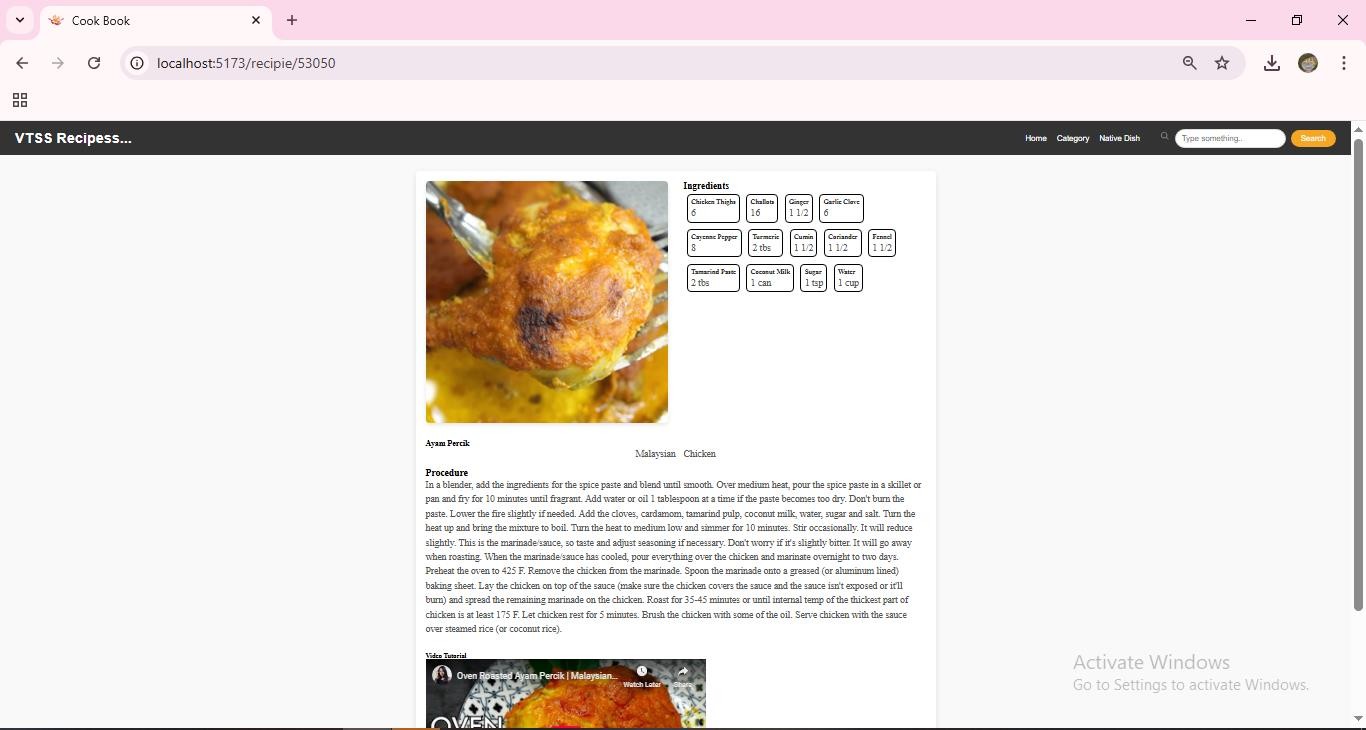
* **Testing Strategy:** Uses **Jest** and **React Testing Library** for unit tests.
* **Code Coverage:** Ensures key components have sufficient test coverage.

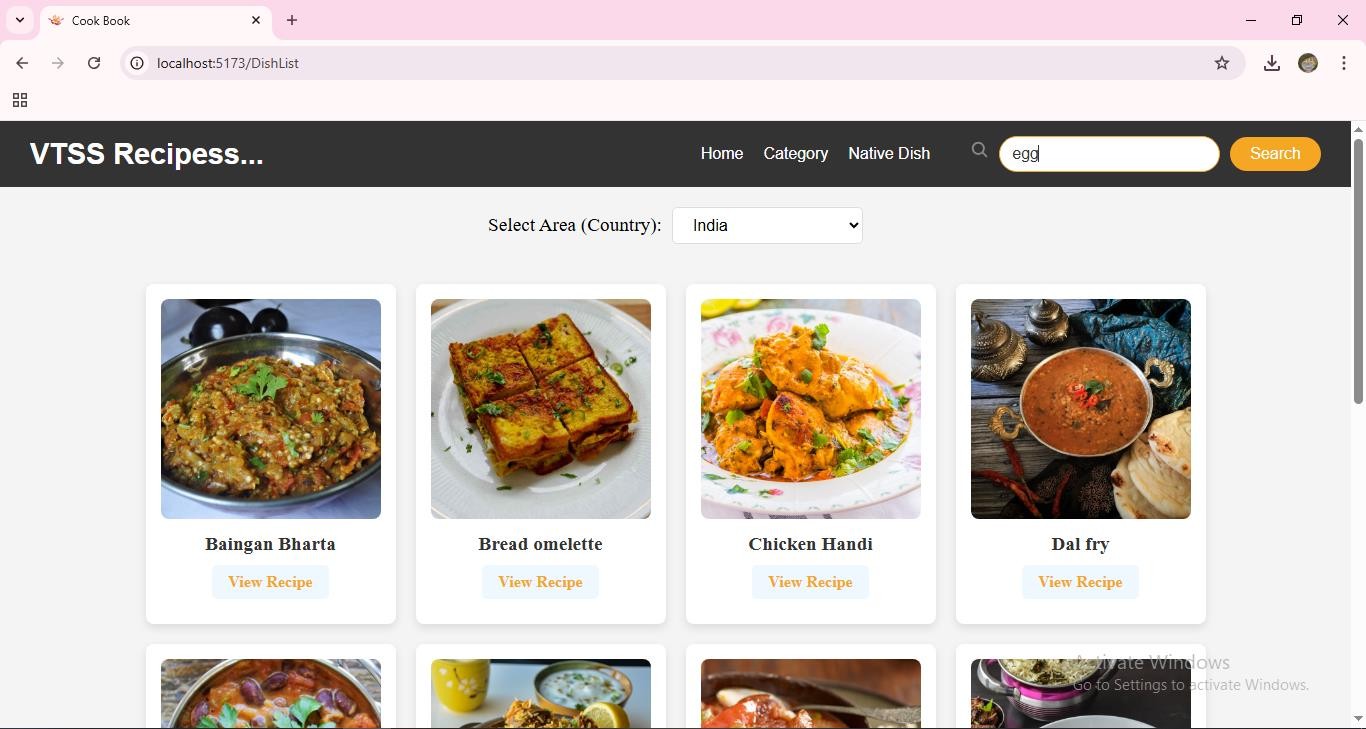
**Screenshots or Demo**

****

**Home Page**

**Cateogory Page**



****

**Demo Link:**

[**https://drive.google.com/file/d/1pLQJp4xJwkfMcZhke4kSkAIHXBl7TkcF/view?usp=drive\_link**](https://drive.google.com/file/d/1pLQJp4xJwkfMcZhke4kSkAIHXBl7TkcF/view?usp=drive_link)

**Known Issues**

* Recipe images are not stored persistently yet.
* Filtering options need refinement.

**Future Enhancements**

* Implement dark mode.
* Add a comments section for each recipe.
* Improve search functionality with category-based filters.