**COOKBOOK**

**Submitted as Part of the Course**

**SEC 160 – FULL STACK DEVELOPMENT -1**

**A logo of a tree

AI-generated content may be incorrect.**

|  |  |  |
| --- | --- | --- |
| **Group Details** | | |
| **Sl. No** | **Reg. No.** | **Name** |
| 1 | AP23110011052 | Ch. Jaswanth |
| 2 | AP23110010703 | B. Losish Ram |
| 3 | AP23110010252 | M. Sesha Sai |
| 4 | AP23110010740 | J. Karthik |

**SRM University-AP**

**December-2025**

**COOKBOOK (React)**

**Introduction:**

Welcome to “Cook Book”, a premium, responsive recipe manager application designed to elevate your culinary journey in the digital age. In a world where culinary inspiration is abundant but often scattered across disparate sources handwritten notes, bookmarked websites, and social media Saves Cook Book offers a centralized, sophisticated sanctuary for your recipes. Built with the cutting-edge power of “React 19”, “Vite”, and “Tailwind CSS”, Cook Book merges beautiful, minimalist design with robust functionality, providing a seamless platform for discovering, creating, and managing your culinary collection.

Cook Book serves as more than just a digital database; it is a modern web application engineered to prioritize user experience and performance. Unlike traditional recipe sites often cluttered with distractions, Cook Book focuses on clarity and utility. It empowers users with a complete suite of tools: a secure authentication system for personalized access, instantaneous real-time search capabilities to find dishes in milliseconds, and a stunning, adaptive dark mode that ensures readability in any lighting environment.

The application bridges the gap between professional kitchen management and home cooking needs. It allows users to not only view recipes but to curate them, offering granular control over ingredients, preparation steps, and categorization. Whether you are a professional chef documenting complex gastronomic creations or a passionate home cook organizing weeknight favourites Cook Book provides the intuitive, high-performance interface necessary to organize your kitchen creativity effectively.

**Scenario-Based Intro:**

Imagine you've just found a moment of inspiration in the kitchen. You want to cook something special but need the perfect recipe You open “Cook Book”.

As you launch the app, you are greeted by a visually stunning interface, perhaps in a soothing dark mode that's easy on the eyes. You quickly search for "chicken" and filter by "Indian" cuisine. Instantly, a curated list of delicious recipes like "Butter Chicken" appears. You click on one, viewing a beautiful layout of ingredients and step-by-step instructions.

Later, you decide to share your own creation. Logging in as an admin, you access the "Create Recipe" feature. You effortlessly add ingredients, steps, and photos. With a single click, your recipe is live, ready for others to discover and favourite. Cook Book makes this entire process fluid, intuitive, and enjoyable

**Target Audience:**

* Home Cooks: Individuals looking for daily meal inspiration and a place to organize their favourite recipes.
* Culinary Enthusiasts: People passionate about exploring new cuisines and sharing their own culinary experiments.
* Professional Chefs: Users who need a digital repository for their recipe collections**.**

**Project Goals and Objectives:**

The primary goal of Cook Book is to offer a robust and aesthetically pleasing platform for recipe management. Our objectives include:

* User-Centric Design: Develop an interface that is intuitive, responsive, and accessible on all devices.
* Efficient Management: Provide full CRUD (Create, Read, Update, Delete) capabilities for recipes, ensuring users have full control over their content.
* Seamless Discovery: Implement advanced search and filtering mechanisms to help users find exactly what they crave in seconds.
* Modern Tech Stack: Leverage the latest web technologies (React 19, Vite, Tailwind CSS) to ensure high performance and maintainability.

**Key Features:**

* Authentication System: Secure Sign Up and Login with persistent user sessions and role-based access control (Admin vs. Normal User).
* Recipe Management (CRUD): Admins can create, edit, and delete recipes with a comprehensive form handling ingredients, steps, and images.
* Advanced Search & Filter: Real-time search by title or ingredients, and instant filtering by cuisine type.
* Favourites System: Users can save their best-loved recipes to a personal "Favourites" collection.
* Dark Mode: A fully functional dark theme that persists across sessions for comfortable viewing in any lighting.
* Responsive Design: A "wow" factor UI that adapts perfectly to mobile phones, tablets, and desktop screens.

**PRE-REQUISITES:**

Here are the key prerequisites for developing a frontend application using React.js:

* **Node.js and npm:**

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the local environment. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side.

* Download: <https://nodejs.org/en/download/>
* Installation instructions: <https://nodejs.org/en/download/package-manager/>

* **React.js:**

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

* **Create a new React app:**

npm create vite@latest

Enter and then type project-name and select preferred frameworks and then enter

* **Navigate to the project directory:**

       cd project-name

   npm install

* **Running the React App:**

With the React app created, you can now start the development server and see your React application in action.

* **Start the development server:**

npm run dev

This command launches the development server, and you can access your React app at <http://localhost:5173> in your web browser.

* **HTML, CSS, and JavaScript**: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

* **Version Control**: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

• **Git**: Download and installation instructions can be found at: <https://git-scm.com/downloads>

* **Development Environment**: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

• **Visual Studio Code**: Download from <https://code.visualstudio.com/download>

**Project Structure:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

* **src/main.jsx:**Entry point of the React application. It locates the root HTML element and renders the App component, usually wrapped with providers or Strict Mode.
* **src/App.jsx:** Root component that defines the main layout and application routing using React Router**.**
* **src/index.css / src/App.css:**Global styling files. index.css handles Tailwind imports and base styles; App.css manages app-wide component styles.
* **Context**
* **src/context/AuthContext.jsx:**Manages global authentication state and exposes login/logout/user data to components**.**
* **src/context/ThemeContext.jsx:**Controls light/dark mode and persists the user’s theme preference.
* **Services**
* **src/services/api.js:**Configures Axios and provides functions for CRUD operations with the backend (JSON-Server).
* **Components**
* **src/components/Header.jsx:**Top navigation bar containing logo, navigation links, theme toggle, and user profile actions**.**
* **src/components/Footer.jsx:**Footer with copyright text and supplementary links.
* **src/components/RecipeCard.jsx:**Reusable card component showing recipe image, title, cook time, and favorite option**.**
* **src/components/RecipeForm.jsx:**Form component used for adding and editing recipes, handling ingredients and steps dynamically.
* **src/components/SearchBar.jsx:**Search component for filtering recipes by name or ingredients in real time.
* **Pages**
* **src/pages/Home.jsx:**Landing page that displays all recipes and integrates search and filter options.
* **src/pages/RecipeDetail.jsx:**Page showing full details of a recipe fetched using the recipe ID from the URL.
* **src/pages/Login.jsx:**Page containing the login form for user authentication.
* **src/pages/AddRecipe.jsx / src/pages/EditRecipe.jsx:**Pages that wrap RecipeForm for creating or editing recipes respectively.
* **src/pages/Favorites.jsx:**Protected page showing recipes marked as favorites by the user.
* **db.json:**JSON-Server mock database storing users, recipes, and related data.

**Milestone 1: Project Setup and Configuration:**

**1. Install required tools and software:**

**Installation of required tools:**

Open the project folder to install necessary tools.

    In this project, we use:

* React Js
* React Icons & Lucide React
* Tailwind CSS
* Axios
* React Hot Toast
* Framer Motion

**For further reference, use the following resources:**

**[** [**https://react.dev/learn/installation**](https://react.dev/learn/installation) **](** [**https://react.dev/learn/installation**](https://react.dev/learn/installation) **)**

**[** [**https://tailwindcss.com/docs/installation**](https://tailwindcss.com/docs/installation) **](** [**https://tailwindcss.com/docs/installation**](https://tailwindcss.com/docs/installation) **)**

**[** [**https://vitejs.dev/guide/**](https://vitejs.dev/guide/) **](** [**https://vitejs.dev/guide/**](https://vitejs.dev/guide/) **)**

**Milestone 2: Web Development:**

**1. Setup React Application:**

* Create React application.
* Configure Routing.
* Install required libraries.

**App.jsx Component:**

**A screen shot of a computer code

AI-generated content may be incorrect.**

* **Code Description:**
* App.css: The CSS file for styling the App component.
* Home from ./pages/Home: The Home component that will be displayed.
* AuthProvider from ./context/AuthContext: A context provider component to manage the application's authentication state.
* The App function is defined, which returns JSX to render the AuthProvider and Routes.
* The Router component wraps the application to enable client-side routing.
* AuthProvider: This component wraps the entire application, providing it with authentication context data (user, login, logout).
* Routes: Defines the mapping structure for valid URLs in the application (/, /recipe/:id, /login).
* The App component is exported as the default export of the module.

**2. Design UI components:**

* Create reusable components (Header, Footer, RecipeCard).
* Implement layout and styling using Tailwind CSS.
* Add navigation structure.

**3. Implement frontend logic:**

* Integration with API endpoints (using Axios).

A screen shot of a computer program

AI-generated content may be incorrect.

* Implement data binding and state management**.**

**4. Admin Functionality:**

* Admin Login: Secure login mechanism that verifies user credentials against the database.
* Role-Based Access: The system checks the isAdmin flag in the user profile to grant or restrict access.
* CRUD Privileges:
* Create: Admins can access the Add Recipe page to publish new culinary content.
* Update: Admins see an Edit button on their recipes to modify details, ingredients, and steps.
* Delete: Admins have exclusive rights to remove recipes from the database.

**Milestone 3: Components & Logic:**

**Auth Context:**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**Code Description:**

* Creates an AuthContext using createContext() to hold the authentication state (user data) and methods.
* Defines an AuthProvider component that wraps around other components to provide context values.
* Uses the useState hook to create a user state variable, initializing it from localStorage to persist sessions.
* Uses the useEffect hook to automatically sync the user state with localStorage whenever it changes.
* Defines helper functions like login, register, and logout to handle user actions and API calls.
* Returns an `AuthContext.Provider` with a value prop containing user , login, register, logout, and isAuthenticated.
* Exports AuthProvider and a custom hook useAuth for easy access in other parts of the application.

**Recipe Card Component:**

**A computer screen shot of text

AI-generated content may be incorrect.**

**Code Description:**

* Imports Link from react-router-dom for navigation to the detailed recipe view.
* Imports icons (Clock, Users, Heart, ArrowRight) from lucide-react for visual enhancements.
* Defines the RecipeCard component which accepts recipe, toggleFavorite, and isFavorite as props.
* Displays the recipe image with an overlay gradient and a "zoom" hover effect.
* Renders a "Heart" button that toggles the favorite status, changing color based on the isFavorite prop.
* Displays recipe details: Title, Description, Prep Time, and Servings count.
* Uses a Link component wrapping the "View Details" button to navigate to /recipe/:id.
* Exports the RecipeCard component as the default export for use in lists and grids throughout the app.

**Recipe Form Component:**

**A screen shot of a computer code

AI-generated content may be incorrect.**

**Code Description:**

* Imports useForm and useFieldArray from react-hook-form to manage complex form state and dynamic fields (ingredients, steps).
* Imports motion from motion/react (Framer Motion) for smooth entry animations.
* Defines the RecipeForm component accepting initialData, onSubmit, and isEdit props for reusability in both "Add" and "Edit" modes.
* Initializes form with default values for title, description, ingredients, steps, etc.
* Uses useFieldArray to handle dynamic adding/removing of ingredients and cooking steps.
* Mapping over ingredientFields and stepFields allows users to interactively add multiple items.
* The onFormSubmit function formats data (e.g., splitting tags string into an array) before calling the parent onSubmit handler.
* Renders a responsive, styled form using Tailwind CSS grid and flexbox, with validation error messages displayed below inputs.
* Exports RecipeForm to be used by the AddRecipe and EditRecipe pages.

**Project Execution:**

After completing the code, run the react application by using the command "npm run dev", as we are using Vite.js.

Here are some of the screenshots of the application:

**Login Page:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Home Page:**

**A screenshot of a food website

AI-generated content may be incorrect.**

**Recipe Detail Page:**

**A screenshot of a food recipe

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

**Favourite Page:**

**A screenshot of a food menu

AI-generated content may be incorrect.**

**Admin Home Page:**

**A screenshot of a food menu

AI-generated content may be incorrect.**

**Admin Create Recipe Page:**

**A screenshot of a recipe form

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

**Project Links:**

**Project demonstration link:**

[**https://drive.google.com/file/d/1ToxpwwKXZvmE9QcejA7Hini6jnbFisKu/view?usp=sharing**](https://drive.google.com/file/d/1ToxpwwKXZvmE9QcejA7Hini6jnbFisKu/view?usp=sharing)

**Project Code Explanation:**

[**https://drive.google.com/file/d/1-PM0j5VkeG4fYkQWWb4gV1i51T086ax-/view?usp=sharing**](https://drive.google.com/file/d/1-PM0j5VkeG4fYkQWWb4gV1i51T086ax-/view?usp=sharing)

**Project Code Files:**

[**https://github.com/Jaswanth-techie/Cookbook-FSD-**](https://github.com/Jaswanth-techie/Cookbook-FSD-)