



CHRIST
(DEEMED TO BE UNIVERSITY)
BANGALORE • INDIA

DEPARTMENT OF AI, ML & DATA SCIENCE

UI/UX Design Fundamentals

Title: RECIPE BOOK

Submitted BY

S. Mehanath (2460645)

s.mehanath@btech.christuniversity.in

Nanda Krishnan (2462116)

nandakrishnan@btech.christuniversity.in

Tom Bobby Alex (2462162)

tom.boby@btech.christuniversity.in

Instructor Name :Mr Narendra

Institution :Christ University Kengeri Campus

Date of Submission :26/09/2025



CHRIST

(DEEMED TO BE UNIVERSITY)

BANGALORE • INDIA

Certificate

*This is to certify that **S Mehanath(2460645),NandaKrishnan(2462116),Tom Boby Alex(2462162)** has successfully completed the Capstone Project work for **UI/UX Design Fundamentals** in partial fulfilment for Btech during the year 2025-2026.*

Name : S Mehanath

Register No. :2460645

Name : Tom Boby Alex

Register No. :2462162

Name : Nanda Krishnan

Register No. :2462116

Abstract

This project is an interactive, single-page recipe book application designed to provide a fluid and dynamic user experience. Built with a combination of HTML, CSS, JavaScript, jQuery, and the Bootstrap 5 framework, it allows users to browse, search, and view a collection of recipes entirely within a single web page, eliminating the need for disruptive page reloads and creating a seamless browsing environment. The application's core logic and content are self-contained, making it a portable and efficient demonstration of modern front-end development techniques.

The application's foundation is a client-side JavaScript array that acts as a local database, holding all the recipe information. Upon loading, the script immediately processes this data to dynamically generate a visually appealing grid of recipe cards. Each card serves as a snapshot, presenting a recipe's image, title, a short description, and key tags like its cuisine and difficulty level. This initial display fulfils the primary requirement of listing all available recipes in an organized and accessible manner.

A key interactive feature is the powerful, real-time search functionality. Users can type into a search bar located in the header to instantly filter the recipes. This search is designed to be comprehensive, as it queries not only the titles of the recipes but also their entire list of ingredients. This allows users to find dishes either by name or by a specific component they might have on hand, making the tool both a recipe browser and a practical cooking assistant.

Beyond the simple text search, the application provides a multi-faceted filtering system for more granular control. Users can refine the displayed recipes using a set of dropdown menus to select a specific **Cuisine**, **Difficulty Level**, or **Dietary Preference** such as Vegetarian or Vegan. These filters can be used individually or in combination with each other and the text search, allowing for highly specific queries. As filters are applied, active selections are displayed as small "chips," giving users clear feedback on their current criteria.

To provide comprehensive information without navigating away from the main grid, the application uses a modal window for its **Recipe Details Page**. Clicking the "View Recipe" button on any card launches this pop-up, which is dynamically populated with the selected recipe's complete information. This detailed view includes an enlarged image, a full list of necessary ingredients, and clear, step-by-step preparation instructions, offering all the necessary information in an easily accessible format.

Finally, the entire user interface is built to be fully **responsive**, ensuring a consistent and navigable experience across all devices. By leveraging Bootstrap's robust grid system, the layout automatically adapts to different screen sizes, from small mobile phones to large desktop monitors. This is complemented by a custom dark-themed aesthetic defined with CSS, which gives the recipe book a modern and polished look while ensuring readability and a pleasant visual experience for the user.

Objectives of the Project

1. Develop a Dynamic Recipe Listing Interface:

- The primary objective is to display a list of all available recipes in a clear, grid-based format. Each recipe card in this listing must visually present the recipe's image, title, and a short description, providing users with an at-a-glance overview of the dish.

2. Implement Comprehensive Search and Filter Functionality:

- The application must feature a robust system to help users find specific recipes. This involves implementing two key mechanisms:
- A text-based search that allows users to look for recipes by their title or ingredients.
- A set of category filters (e.g., dropdown menus) that enable users to narrow down the list by cuisine, difficulty level, and dietary preference.

3. Create an Interactive Recipe Details Page:

- Upon clicking a specific recipe, the application must present a detailed view. This objective is to create an interactive pop-up or modal that dynamically displays the selected recipe's complete information, including a full list of ingredients and step-by-step preparation instructions.

4. Ensure a Fully Responsive Design:

- The project must be easily navigable and visually appealing on all common devices. A key objective is to use the Bootstrap framework to build a fluid and responsive layout that provides an optimal user experience on both mobile and desktop screens.

5. Utilize the Specified Technology Stack:

- The final objective is to build the entire application using the prescribed technologies. This means using HTML for the core structure, CSS for custom styling and theming, and JavaScript with jQuery to handle all interactivity, data filtering, and dynamic content rendering.

6. Enhance User Experience with Intuitive Feedback:

- The objective is to create a more intuitive and user-friendly interface by providing clear visual feedback. This includes
- Displaying all currently active search and filter criteria as removable "chips" so the user always knows what they are searching for.
- Implementing a distinct "empty state" message that clearly informs the user when no recipes match their selected filters, preventing confusion.

7. Implement a Scalable and Dynamic Filter System:

- To make the application maintainable and scalable, an objective is to dynamically generate filter options from the recipe data itself.
- Specifically, the Cuisine filter's dropdown menu should not be hard-coded but should be populated automatically from the cuisines present in the JavaScript recipe array.
- This ensures that adding a new recipe with a new cuisine automatically updates the filter options.

8. Structure Code for Maintainability and Clarity:

- A crucial objective is to write clean, well-organized code that is easy to understand and modify in the future.
- This involves separating the JavaScript logic into distinct, single-responsibility functions (e.g., a function for rendering the grid, another for applying filters, and another for handling the details modal). This modular approach improves readability and makes debugging and future enhancements more efficient.

Scope of the Project

1. Core Recipe Database

- **Current Utilization:** The project serves as a self-contained, static digital cookbook. It's perfect for a personal collection of favorite recipes, a portfolio demonstration, or a small, curated menu for a restaurant's website.
- **Potential for Improvement:** Integrate a backend database (like Firebase or Supabase). This would allow recipes to be added, edited, and deleted through a user-friendly interface without needing to alter the source code, transforming it into a dynamic Content Management System (CMS).

2. Data Sourcing

- **Current Utilization:** Being entirely client-side with a JavaScript array as its data source, the app is extremely fast and works offline after the initial load. It requires no complex setup or hosting.
- **Potential for Improvement:** Connect to a third-party recipe API (e.g., TheMealDB, Spoonacular). This would grant users access to thousands of recipes, provide a constantly changing library, and significantly expand the application's content.

3. Search and Filter Functionality

- **Current Utilization:** The system offers an excellent real-time filtering experience for its predefined data set, allowing users to instantly narrow down choices by cuisine, difficulty, or ingredients.
- **Potential for Improvement:** Enhance the search with advanced features like multi-select filters (e.g., choosing "Vegan" and "Gluten-Free" simultaneously), the ability to exclude certain ingredients, and implementing a "fuzzy search" to handle spelling mistakes.

4. User Personalization

- **Current Utilization:** The application currently provides a uniform, public experience where all content is available to every visitor. There is no concept of individual users.
- **Potential for Improvement:** Introduce a complete user authentication system. This would unlock personalization features like saving favorite recipes, creating custom collections ("My Weeknight Dinners"), and uploading private recipes.

5. Community and Social Engagement

- **Current Utilization:** It functions as a read-only portal for information consumption. There are no features for user interaction or community building.
- **Potential for Improvement:** Incorporate social features such as a star-based rating and review system, a comments section for users to share tips on each recipe, and buttons to easily share recipes on social media platforms.

6. Content and Feature Expansion

- **Current Utilization:** The project's scope is strictly defined around recipes, focusing on ingredients and preparation steps.
- **Potential for Improvement:** Broaden the application's utility by adding a meal planning feature with a weekly calendar, a tool to automatically generate a shopping list from selected recipes, or a blog section for cooking tips and articles.

7. Interactive Cooking Assistance

- **Current Utilization:** Recipes are presented in a static, readable format that the user follows on their own.
- **Potential for Improvement:** Add a "Cook Mode" that displays instructions one step at a time in a large, easy-to-read font. This mode could include integrated kitchen timers for specific steps and a feature to automatically scale ingredient quantities based on serving size.

8. Technology Stack Modernization

- **Current Utilization:** The project is an excellent example of foundational web technologies, using jQuery for DOM manipulation.
- **Potential for Improvement:** Refactor the entire application using a modern JavaScript framework like **React**, **Vue**, or **Svelte**. This would create a more

maintainable, scalable, and performant application built on a component-based architecture.

9. Nutritional Information

- **Current Utilization:** The dietary information is limited to a single, simple tag (e.g., "Vegetarian").
- **Potential for Improvement:** Integrate with a nutrition API to automatically calculate and display detailed nutritional data (calories, protein, fats, carbs) for each recipe, providing valuable health insights to users.

10. Accessibility (a11y)

- **Current Utilization:** It includes basic accessibility features like image alt text and form labels.
- **Potential for Improvement:** Conduct a full accessibility audit. This would involve implementing ARIA roles for complex widgets, ensuring high colour contrast for all text, and guaranteeing full keyboard navigability for all interactive elements.

11. Advanced Deployment

- **Current Utilization:** As a single static file, it can be deployed for free on services like GitHub Pages or Netlify with zero configuration.
- **Potential for Improvement:** If backend features are added, the project could be deployed as a full-stack application on a cloud platform like AWS or as a serverless application, enabling greater scalability and power.

12. Monetization Pathways

- **Current Utilization:** The project is currently non-commercial, serving as a personal tool or portfolio piece.
- **Potential for Improvement:** Explore monetization strategies, such as creating a "premium" section with exclusive recipes for subscribers, adding affiliate links for purchasing kitchen tools or specific ingredients, or integrating with grocery delivery services.

Tools & Technologies Used

The development of the Recipe Book involves the use of several essential tools and technologies to ensure a structured, responsive, and user-friendly design. Each tool serves a specific purpose in the creation, styling, testing, and debugging of the website:

1. **HTML5** – Used for structuring the web pages and organizing content. HTML5 provides semantic elements that make the website more readable and accessible,

allowing proper headings, sections, tables, and forms for effective presentation of information.

2. **CSS3** – Employed for styling the website, including layouts, colours, fonts, spacing, and responsive design. CSS3 allows the implementation of flexible designs, animations, and hover effects to enhance the visual appearance and usability of the website.
3. **Bootstrap** – A front-end framework used to create responsive, mobile-friendly layouts quickly and efficiently. Bootstrap’s prebuilt classes and grid system ensure consistent design across different devices and screen sizes.
4. **VS Code** – The primary code editor used for writing and managing the HTML, CSS, and JavaScript code. VS Code provides features like syntax highlighting, code suggestions, and extensions that streamline the development process.
5. **Chrome DevTools** – Utilized for testing, debugging, and inspecting the website. DevTools helps in identifying layout issues, optimizing performance, and ensuring cross-browser compatibility.
6. **Open-Source Tools & Resources** – The project relies entirely on open-source tools and libraries to maintain cost-effectiveness and ensure ease of maintenance and scalability in the future.
7. **jQuery Library** –The project heavily relies on **jQuery**, an open-source JavaScript library, to simplify interactions with the HTML document. Instead of writing lengthy vanilla JavaScript, jQuery was used for its concise syntax to:

Select and manipulate DOM elements (e.g., `$('#grid').empty()`).

Handle user events like clicks and input changes (e.g., `$('#search').on('input', ...)`).

Easily retrieve values from form fields (e.g., `$('#dietFilter').val()`).
8. **Content Delivery Networks (CDNs)** –To optimize performance and reduce server load, the project does not host the Bootstrap and jQuery files locally. Instead, it utilizes public **CDNs (Content Delivery Networks)** like `cdn.jsdelivr.net` and `code.jquery.com`. These open networks distribute the library files from servers located globally, ensuring faster load times for users anywhere in the world.
9. **Version Control System (Git & GitHub)** **Git** – the open-source distributed version control system, was essential for managing the project's source code. It allowed for tracking every change, experimenting with new features in separate branches, and rolling back to previous versions if needed. **GitHub**, a platform built around Git, was used to host the code repository, providing a backup and a platform for potential collaboration.
10. **Royalty-Free Image Repositories** –A recipe book is a highly visual project. All the images used in the recipe data were sourced from open-source and royalty-free platforms like **Unsplash**. These resources provide high-quality, professional photographs that can be used for free, which is crucial for creating a visually appealing application without incurring licensing costs.

11. **Modern JavaScript Standards (ES6)** – The project's JavaScript code adheres to the modern **ECMAScript 2015 (ES6)** standard. This open standard allows for more robust and readable code by using features like:

- “const” for declaring constants, preventing accidental reassignment.
- Arrow functions (=>) for a more concise way to write functions.

12. **Live Server for Local Development** –Within VS Code, the open-source **Live Server** extension was used extensively. This tool spins up a local development server with a live reload feature, automatically refreshing the browser whenever a code file is saved. This provides immediate feedback and significantly speeds up the process of styling, testing, and debugging the application.

Technology Used	Purpose
HTML5	Markup and content structure
CSS3	Styling and layout management
VS Code	Code editor
Chrome DevTools	Testing and debugging
Bootstrap	A front-end framework
Java script	Makes websites interactive and dynamic
J Query	Document traversal and manipulation, event handling

HTML Structure Overview

The Recipe Book is built as a **single-page application (SPA)** using HTML5, with a structure designed for clarity and dynamic content injection.

1. Semantic Component-Based Structure:

The application utilizes semantic tags like <nav>, <main>, <section>, and <footer> to define the primary layout regions. The structure is organized into logical components:

- A <nav> for the header, brand, and search bar.
- A <section> for all filtering controls (Cuisine, Difficulty, Diet).

- A `<main>` area that serves as the container for the dynamically generated recipe grid.
- A Bootstrap Modal `<div>` which acts as the "Recipe Details Page".

2. Dynamic Content Holders:

Key elements like `<div id="grid">` and `<div id="recipeModal">` are designed as empty containers in the initial HTML. They serve as targets for JavaScript to dynamically render recipe cards and populate recipe details, which is central to the application's interactive nature.

3. Data Attributes for Interactivity

To link the user interface with the JavaScript data, **HTML data attributes** (e.g., `data-id="..."` on buttons) are used. These attributes store unique recipe identifiers directly in the HTML, providing a simple and effective way for the JavaScript to know which recipe's details to display when a button is clicked.

4. Content Hierarchy:

A clear content hierarchy is maintained using heading tags (`<h5>` for card titles, `<h6>` for modal subheadings), paragraphs (`<p>`), and lists (`` and `` for ingredients and steps). This ensures that all information, whether on the cards or in the modal, is presented logically and is easy for users to scan.

CSS Styling Strategy :

The project's styling is implemented directly within the HTML file using a `<style>` block, with a focus on a modern, dark-themed aesthetic and maintainability through CSS variables.

1. Internal CSS & Variable-Based Theming:

All styles are managed in an internal stylesheet. The core of the design is a custom theme built with **CSS Variables** (e.g., `--bg`, `--panel`, `--accent`). This approach allows for easy and consistent theming; changing a single variable's value can update the entire application's colour scheme.

2. Framework-Driven Layouts:

The layout relies heavily on the **Bootstrap 5 framework**.

- **Bootstrap Grid** (`row`, `col-12`, `col-sm-6`, `col-lg-4`) is used to create the main responsive grid for the recipe cards, ensuring it adapts perfectly across all device sizes.
- **Flexbox Utilities** (`d-flex`, `flex-column`, `mt-auto`) are used within the cards to vertically align content and push the "View Recipe" button to the bottom, ensuring a uniform card appearance regardless of content length.

3. Custom Component Styling :

Beyond Bootstrap, custom CSS is applied to create a unique look for components like recipe cards, form inputs, and informational "tags" or "chips." These styles override or supplement Bootstrap's defaults to achieve the desired dark-mode aesthetic.

Challenges Faced & Solutions :

During development, several challenges specific to creating a dynamic, client-side application were addressed.

1. Managing Application State with jQuery:

- **Challenge:** Without a state management library (like React State), keeping track of multiple active filters (search term, cuisine, difficulty, diet) and ensuring the UI is always in sync can become complex.
- **Solution:** A single, centralized `applyFilters()` function was implemented. This function acts as the single source of truth; it reads the current values from **all** filter inputs, applies the combined logic to the master recipe array, and triggers a complete re-render of the recipe grid.

2. Handling Events on Dynamically Created Elements:

- **Challenge:** The recipe cards and their "View Recipe" buttons do not exist on page load; they are generated by JavaScript. A standard `$('.view-btn').click()` event listener would fail to attach to them.
- **Solution: Event Delegation** was used. A single event listener was attached to the static parent container (`#grid`). This listener waits for a click to "bubble up" and then checks if the clicked element matches the `.view-btn` selector, allowing it to efficiently manage events for any number of dynamically added buttons.

3. Populating Modal Content Dynamically:

- **Challenge:** The content of the recipe details modal must change completely depending on which of the many recipe cards is clicked.
- **Solution:** The `data-id` attribute on each button was used as a key. When a button is clicked, its `data-id` is retrieved. This ID is used to find the correct recipe object in the JavaScript array. A dedicated `openModal()` function then takes this object and injects its properties (title, image, ingredients, steps) into the appropriate elements within the modal.

4. Providing Clear User Feedback on Active Filters:

- **Challenge:** Users can easily lose track of which filters they have applied, leading to confusion about why certain recipes are or are not being displayed.
- **Solution:** A dedicated UI section was created to display "active filter chips." A separate JavaScript function dynamically generates these chips based on the current filter selections, providing an immediate and clear visual summary. Each chip includes a "remove" icon, allowing users to easily undo a filter.

5. Gracefully Handling "No Results" Scenarios:

- **Challenge:** A blank screen after applying filters that yield no matches is an unhelpful and poor user experience.
- **Solution:** An "empty state" container was designed and included in the HTML. The core rendering function checks the length of the filtered recipe array. If it is empty, the recipe grid is hidden and this "empty state" message is displayed, informing the user and suggesting they try different filter criteria.

Outcome

The successful completion of this project yields a fully functional, self-contained, and interactive single-page Recipe Book application. The final product achieves the following key outcomes:

1. A Dynamic and Responsive User Interface:

A visually appealing application that provides a seamless experience on desktops, tablets, and mobile devices, built on the Bootstrap 5 framework.

2. A Centralized Recipe Listing:

An organized grid view that dynamically displays all recipes from a client-side data source, with each recipe card showing its image, title, description, and key tags.

3. Real-Time Search and Filtering:

A powerful and instantaneous filtering system that allows users to narrow down the recipe list by typing in a search query (for titles/ingredients) or selecting categories like Cuisine, Difficulty, and Dietary Preference.

4. An Interactive Details View:

A modal-based "Recipe Details Page" that provides comprehensive information, including a full ingredient list and step-by-step preparation instructions, without requiring a page reload.

5. Enhanced User Feedback:

An intuitive interface that provides clear feedback to the user, such as displaying active filter "chips" and showing a helpful message when no recipes match the selected criteria.

6. A Complete Portfolio Piece:

A polished project that effectively demonstrates proficiency in front-end web development using core technologies like HTML5, CSS3, JavaScript, jQuery, and Bootstrap.

Future Enhancements

While the current project is a complete application, it is built on a foundation that allows for numerous exciting improvements. The following enhancements could be implemented in the future:

1. Backend & Database Integration:

- Connect the application to a real-time backend service like **Firebase** or **Supabase**. This would allow for a persistent database, enabling users to add, edit, and delete recipes through a dedicated admin panel.

2. User Accounts and Personalization:

- Implement a user authentication system where users can sign up and log in. This would enable personalized features like saving recipes to a personal "Favorites" or "Cookbook" collection.

3. Community and Social Features:

- Introduce a **rating and review system** for users to share their feedback and tips on recipes.
- Add social sharing buttons to allow users to easily post their favorite recipes to platforms like Pinterest, Facebook, or Twitter.

4. API Integration for Expanded Content:

- Integrate with a third-party recipe API (like TheMealDB or Spoonacular) to fetch and display thousands of recipes from an external source, dramatically increasing the application's content.

5. Interactive "Cook Mode":

- Develop a special viewing mode designed for use in the kitchen. This "Cook Mode" would display preparation steps one at a time in a large, easy-to-read font and could prevent the device's screen from dimming.

6. Ingredient Scaling and Shopping Lists:

- Add a feature to automatically adjust ingredient quantities based on the desired number of servings.
- Allow users to select multiple recipes and generate a consolidated shopping list that they can save or print.

7. Modern Framework Refactoring:

- Rebuild the application using a modern JavaScript framework such as **React**, **Vue**, or **Svelte**. This would improve state management, performance, and the overall maintainability of the codebase.

8. Advanced Nutritional Information:

- Integrate with a nutrition API to automatically fetch and display detailed nutritional data (calories, macros, vitamins) for each recipe.

Sample code

HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>Mini Recipe Card</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>

  <div class="container">
    <h1>Recipe Book</h1>
    <div id="recipeContainer"></div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

CSS:

```
body {  
  background: #0f1420;  
  color: #e8eefc;  
  font-family: Arial, sans-serif;  
  margin: 2rem;  
}  
  
.container {  
  max-width: 600px;  
  margin: auto;  
}  
  
.card {  
  background: #151c2f;  
  border: 1px solid #273556;  
  border-radius: 10px;  
  padding: 1rem;  
  margin-top: 1rem;  
}  
  
.card img {  
  width: 100%;  
  border-radius: 6px;  
}  
  
.card h2 {  
  color: #ffd7c2;  
}
```

JAVA SCRIPT:

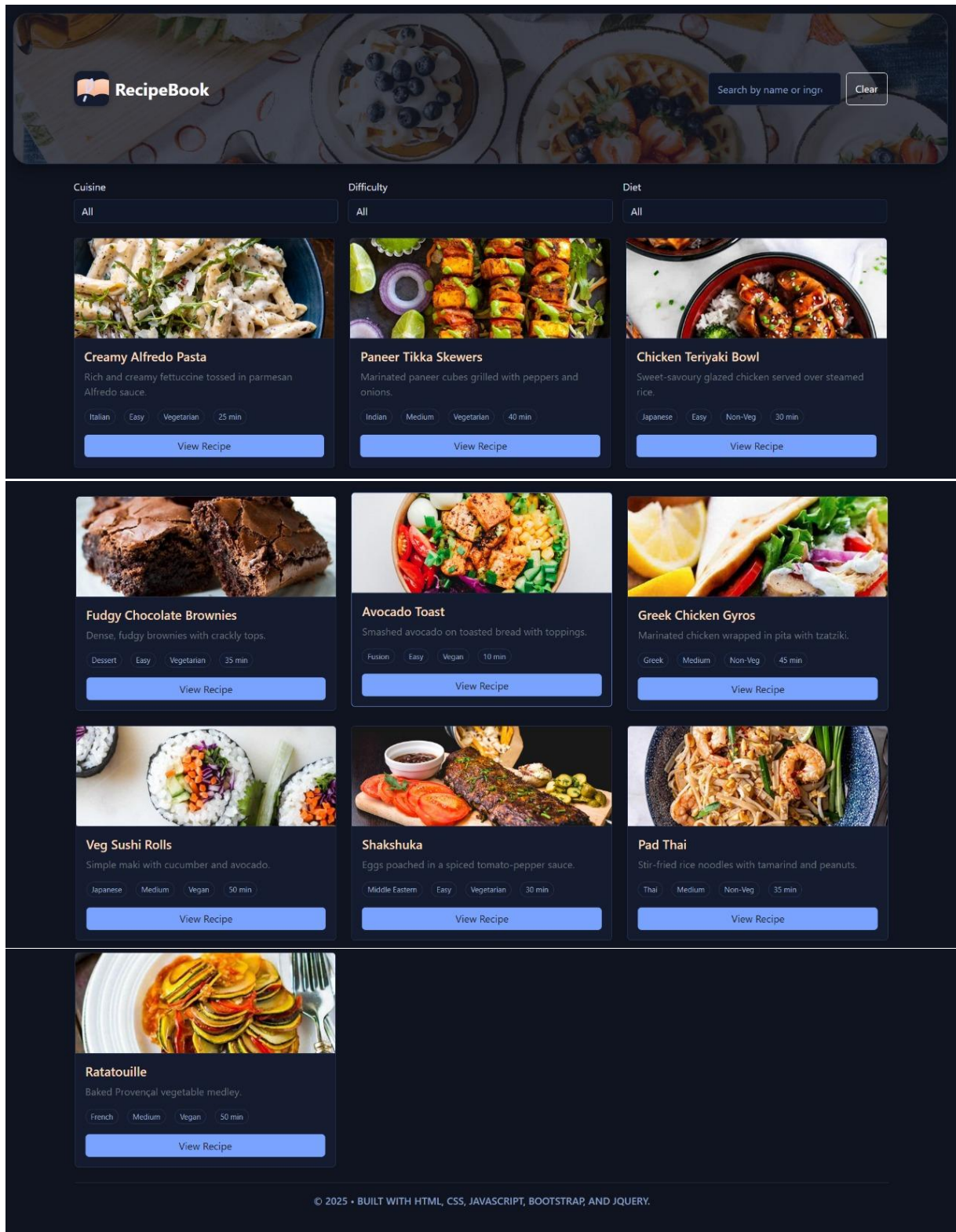
```
const recipe = {
  title: "Avocado Toast",
  image: "https://images.unsplash.com/photo-1546069901-ba9599a7e63c?q=80&w=1200&auto=format&fit=crop",
  description: "Smashed avocado on toasted bread with toppings."
};

const container = document.getElementById("recipeContainer");

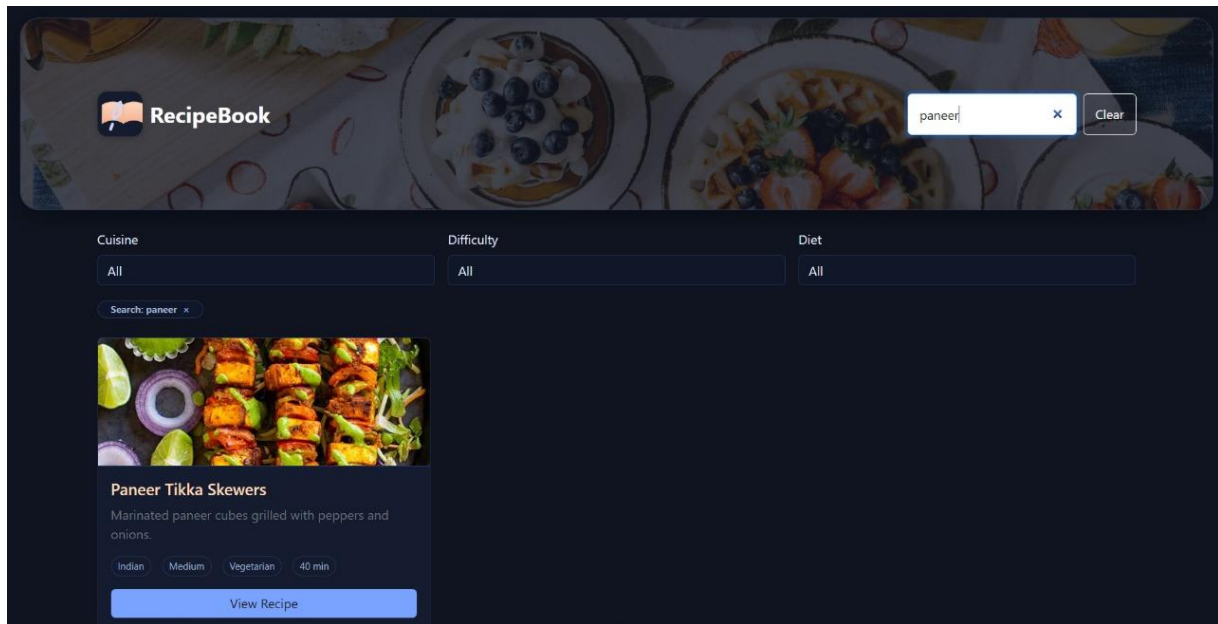
container.innerHTML = `
  <div class="card">
    
    <h2>${recipe.title}</h2>
    <p>${recipe.description}</p>
  </div>
`;
```


Output/ScreenShot

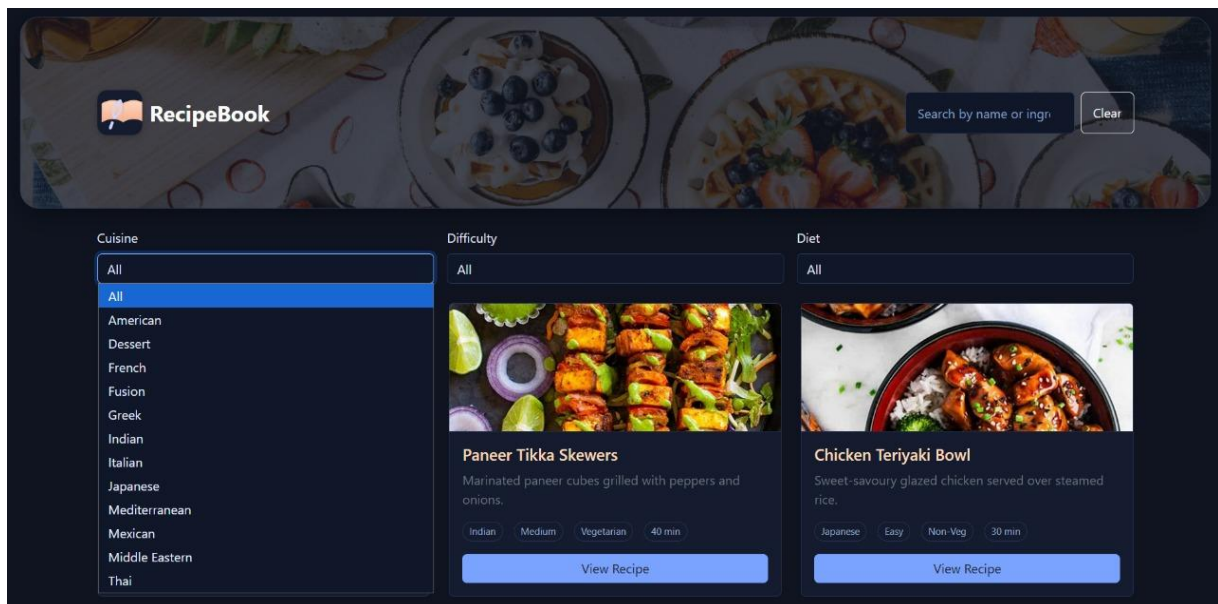
Whole Website overview:



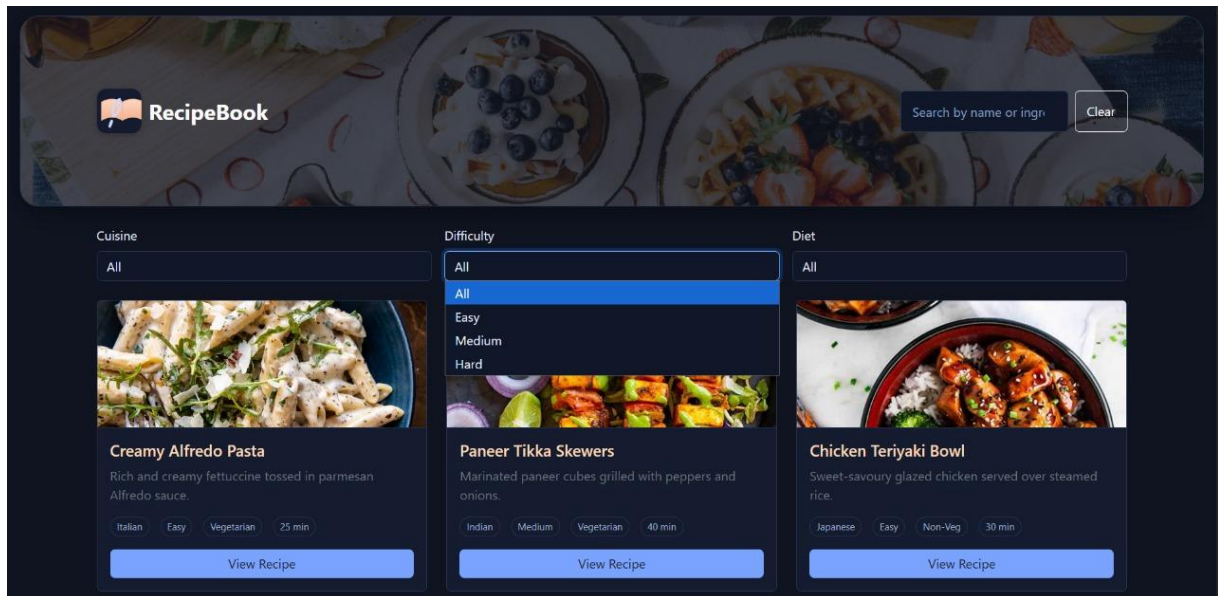
Searching for recipes using a specific ingredient:



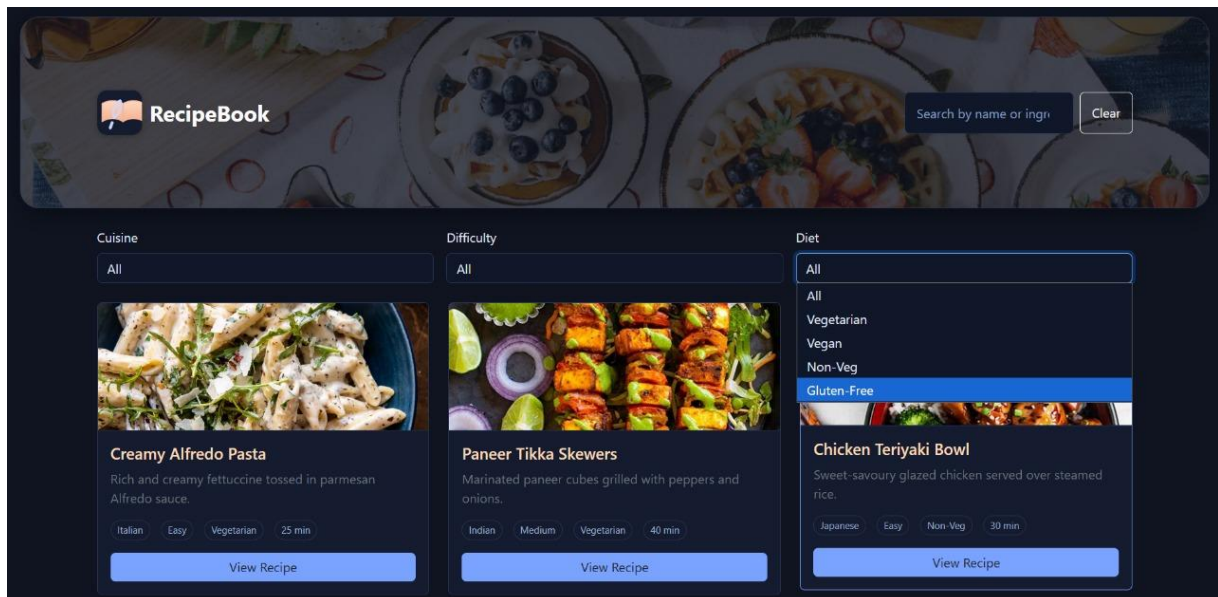
Browsing the available cuisine-based filter options:



Filtering the recipe list by difficulty level:




Selecting a specific dietary preference to filter recipes:



Viewing a recipe's detailed ingredients and steps

Paneer Tikka Skewers



Indian Medium Vegetarian 40 min

Marinated paneer cubes grilled with peppers and onions.

Ingredients	Steps
<ul style="list-style-type: none">• Paneer• Yogurt• Ginger-garlic paste• Tandoori masala• Lemon• Capsicum	<ol style="list-style-type: none">1. Whisk yogurt with spices and lemon.2. Marinate paneer & veggies 20–30 min.3. Skewer alternately.4. Grill or pan-sear until charred.5. Serve with mint chutney.

Close

Conclusion

The Interactive Recipe Book project successfully achieved its primary objective of developing a fully functional, responsive, and user-friendly single-page application. By adhering to the project statement, all core requirements were met, resulting in a seamless platform for users to browse, search, and view a curated collection of recipes. The project effectively demonstrates the practical application of foundational front-end technologies to create a modern and dynamic web experience.

Through the strategic use of **HTML5**, **CSS3**, **JavaScript**, **jQuery**, and the **Bootstrap 5 framework**, the application delivers key features including a dynamic recipe grid, real-time search and filtering capabilities, and an interactive modal for detailed recipe views. The development process provided valuable insights into overcoming common challenges in client-side web applications, such as managing UI state without a dedicated framework, handling events on dynamically generated content, and ensuring a consistent user experience across all device sizes.

The final product stands as a robust and polished portfolio piece. It not only fulfills all initial requirements but also establishes a solid foundation for future enhancements, such as backend integration for a dynamic database, user authentication for personalized content, and the addition of community features. Ultimately, the Recipe Book project is a testament to a comprehensive understanding of web development principles, culminating in an application that is both technically sound and highly engaging for the end-user.

Reference

L&T LMS : <https://learn.lntedutech.com/Landing/MyCourse>

Chat,GPT: <https://chat.openai.com>