Project Documentation

CUSTOMIZED AI KITCHEN FOR INDIA

Intel® Unnati Industrial Program 2024

Team Members:

1. KOWSHIK- Team Lead and Front end Developer
2. THOUFIQ AHAMED .A- Back end Developer Lead
3. DHANUSRI.T - Project Managing Lead and Documentation Assist
4. ANISH FATHIMA.N - Documentation Lead and Testing

Faculty Mentor :

1. P.VALARMATHI

Table of Contents:

1. Introduction
2. Project Overview
3. Features
4. System Architechture
5. Technologies Used
6. Set Up Instruction
7. Functionalities and Implementation
8. Future Enhancements
9. Algorithm
10. Conclusion
11. References

1. Introduction

The AI-Powered Recipe Generator is an innovative web application designed to revolutionize the way you plan and prepare meals. By leveraging advanced API technology and a user-friendly interface, this application allows users to discover recipes based on the ingredients they already have at home. Whether you're a culinary novice or an experienced home cook, this tool can inspire creativity and streamline your cooking process.

At its core, the application connects to TheMealDB API, a rich database of meal recipes, categories, and detailed instructions. Users can search for recipes by entering specific ingredients they have on hand or by selecting a category from a diverse range of meal types, such as breakfast, seafood, or vegetarian options. The AI-Powered Recipe Generator then presents a list of recipes that match the search criteria, complete with images and links to detailed cooking instructions.

What sets this application apart is its focus on personalization and ease of use. The clean, responsive design ensures a seamless experience across all devices, from desktops to smartphones. Users can quickly find recipes that suit their tastes and dietary preferences, view detailed instructions, and even watch YouTube videos for step-by-step guidance on preparing the dishes.

This project aims to simplify meal planning, reduce food waste by making use of available ingredients, and inspire users to try new and exciting recipes. By integrating modern web technologies like HTML, CSS, and JavaScript, the AI-Powered Recipe Generator provides an engaging and efficient way to enhance your culinary journey.

2. Project Overview

The project addresses the challenge of simplifying recipe discovery and cooking guidance through an AI-powered web platform. Users can input either a category (starter, vegan, vegetarian) or a specific main ingredient (e.g., egg, chicken, paneer) that they have available. The system then generates relevant recipes based on the input, providing detailed cooking instructions and, when available, links to cooking videos on YouTube for visual assistance.

Solution Description

The solution is a user-friendly website for recipe generation that offers the following features:

- **Input Options:** Users can enter a category (e.g., starter, vegan, vegetarian) or specify a main ingredient (e.g., egg, chicken, paneer) they wish to use in their recipe.
- **Recipe Generation:** Utilizes AI algorithms to match user input with a database of recipes, ensuring relevance and diversity in the suggested dishes.

- **Detailed Instructions:** Provides comprehensive cooking instructions, ingredient lists, and nutritional information for each recipe.
- **Video Integration:** Offers an option to view cooking videos related to selected recipes, directing users to YouTube for additional guidance.

Key Features

1. **User-Friendly Interface:** Simple and intuitive design for easy navigation and input.
2. **AI-Driven Recommendation:** Harnesses machine learning to suggest recipes based on user preferences and available ingredients.
3. **Comprehensive Recipe Details:** Displays thorough recipe details including cooking time, difficulty level, and serving size.
4. **Video Tutorial Access:** Links to relevant cooking videos on YouTube to assist users with visual instructions when available.

Goals

- Enhance user experience in discovering and preparing diverse recipes based on available ingredients or dietary preferences.
- Provide clear and concise cooking instructions to assist users of all skill levels.
- Promote culinary exploration by suggesting recipes tailored to specific user inputs and preferences.

Benefits

- Empowers users to create meals efficiently using ingredients they have on hand.
- Encourages healthier eating habits with access to nutritional information and diverse recipe options.
- Facilitates learning and skill development in cooking through integrated video tutorials.

3. Features

The AI-Powered Recipe Generator offers a variety of features designed to enhance the user experience and make meal planning both efficient and enjoyable. Here's a closer look at its key functionalities:

1. Ingredient-Based Recipe Search:

Users can input one or more ingredients they have on hand to find recipes that incorporate these items.

The application fetches relevant recipes from TheMealDB API, ensuring that users can make the most of their available ingredients.

2. Category Selection:

A dropdown menu allows users to select a category of meals, such as "Breakfast," "Seafood," or "Vegetarian."

This feature helps users narrow down their search to specific types of dishes, catering to different tastes and dietary preferences.

3. Search Results Display:

The application displays a list of recipes that match the user's search criteria, each with a thumbnail image and the recipe name.

This visual presentation makes it easy for users to browse and select recipes.

4. Detailed Recipe View:

Clicking on a recipe reveals detailed information, including the recipe title, category, and step-by-step cooking instructions.

An image of the finished dish is also provided, giving users a visual reference for what they're preparing.

5. Video Instructions:

Each detailed recipe view includes a link to a YouTube video, offering users a step-by-step visual guide on how to prepare the meal.

This feature is particularly helpful for users who prefer video instructions over text.

6. Responsive Design:

The application is designed to be fully responsive, ensuring a seamless experience on all devices, whether it's a desktop, tablet, or smartphone.

Users can search for recipes and view instructions from the convenience of any device.

7. User-Friendly Interface:

The clean and intuitive interface ensures that users can easily navigate through the application, from searching for recipes to viewing detailed instructions.

Interactive elements such as buttons and dropdown menus are designed for ease of use.

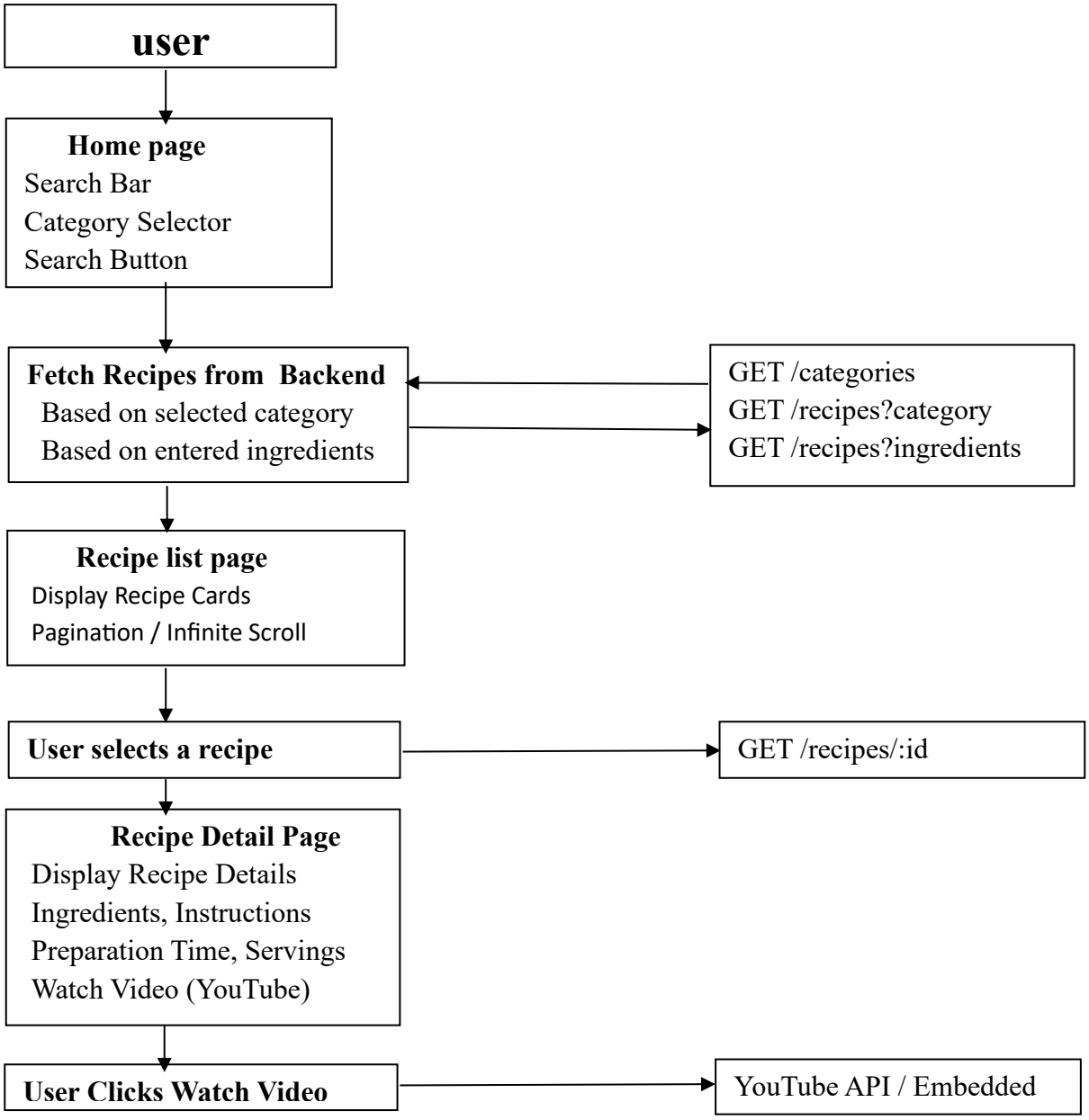
8. Minimalist and Aesthetic Design:

The application features a modern and minimalist design, with a focus on readability and aesthetic appeal.

The use of colors, fonts, and spacing enhances the overall user experience, making it enjoyable to use.

By combining these features, the AI-Powered Recipe Generator aims to provide a comprehensive and enjoyable tool for anyone looking to explore new recipes and make the most of their available ingredients.

4. System Architechture



5. Technologies Used

The AI-Powered Recipe Generator leverages a combination of modern web development technologies to deliver a robust and user-friendly experience. Here's an in-depth look at the key technologies employed in this project:

1. HTML (HyperText Markup Language):

HTML is the backbone of the application, providing the essential structure and content of the web pages.

It defines the layout, including headings, paragraphs, buttons, and input fields, ensuring a well-organized and accessible interface.

2. CSS (Cascading Style Sheets):

CSS is used to style the HTML elements, giving the application a visually appealing and cohesive look.

The project utilizes custom styles and incorporates external libraries such as Font Awesome for icons and Google Fonts for typography.

Media queries and responsive design techniques ensure that the application looks great on all devices, from desktops to smartphones.

3. JavaScript:

JavaScript adds interactivity and dynamic functionality to the application.

The script handles events like button clicks, fetching data from APIs, and updating the DOM (Document Object Model) to display search results and detailed recipe information.

It enhances the user experience by enabling real-time data fetching and seamless content updates without requiring page reloads.

4. TheMealDB API:

TheMealDB API provides a rich database of meal recipes, categories, and detailed instructions.

The application fetches data from this API to display relevant recipes based on user input, ensuring a diverse and extensive selection of meal options.

API endpoints are used to retrieve categories, filter recipes by ingredients, and fetch detailed recipe information, including images and YouTube video links.

5. Font Awesome:

Font Awesome is an icon library used to enhance the visual appeal and usability of the application.
Icons such as search buttons, recipe links, and close buttons provide intuitive visual cues, making the interface more user-friendly.

6. Google Fonts:

The application utilizes Google Fonts to ensure a clean and modern typography.
The chosen fonts, such as 'Poppins', enhance readability and contribute to the overall aesthetic of the application.

7. Responsive Design Techniques:

The application employs responsive design techniques to ensure a seamless user experience across various devices and screen sizes.
CSS media queries adjust the layout and elements to fit different screen dimensions, providing an optimal viewing experience on desktops, tablets, and smartphones.
By combining these technologies, the AI-Powered Recipe Generator achieves a balance of functionality, performance, and visual appeal, delivering a comprehensive and engaging tool for users to discover and prepare new recipes.

6. Setup Instructions

Setting up the AI-Powered Recipe Generator is straightforward and requires only a few steps. Follow the instructions below to get the application up and running on your local machine:

1. Prerequisites:

Ensure you have a modern web browser installed (e.g., Google Chrome, Mozilla Firefox, Safari).

Basic understanding of HTML, CSS, and JavaScript is helpful but not mandatory.

2. Clone the Repository:

Start by cloning the project repository from your preferred version control system (e.g., GitHub). If the repository is on GitHub, use the following command.

git clone https://github.com/your-username/ai-powered-recipe-generator.git

Navigate to the project directory: **cd ai-powered-recipe-generator**

3. Set Up Project Structure:

Ensure the project directory contains the following files and folders:

```
ai-powered-recipe-generator/
├── index.html
├── style.css
├── script.js
└── images/
```

4. HTML File:

The index.html file is the main entry point of the application. It contains the basic structure and references to the CSS and JavaScript files.

Open the index.html file in your preferred code editor to review and make any necessary adjustments.

5. CSS File:

The style.css file contains all the styling rules for the application. Ensure that it is properly linked in the <head> section of your index.html file:

<link rel="stylesheet" href="style.css">

6. JavaScript File:

The script.js file contains the functionality and interactivity of the application. It should be linked at the bottom of the index.html file before the closing </body> tag:

<script src="script.js"></script>

7. External Resources:

Ensure the necessary external resources are included in the index.html file, such as Font Awesome and Google Fonts:

<linkrel="stylesheet"href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css">

<linkhref="https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700;800;900&display=swap" rel="stylesheet">

8. Running the Application:

To run the application, simply open the index.html file in your web browser. You can do this by double-clicking the file or right-clicking and selecting "Open with" followed by your browser of choice.

Alternatively, you can use a local development server (e.g., Live Server extension for Visual Studio Code) for a better development experience.

npx live-server - This will automatically open the application in your default web browser and reflect any changes you make in real-time.

9. Testing the Application:

Once the application is running, test its functionality by entering ingredients or selecting categories to search for recipes.

Ensure that the search results display correctly, and detailed recipe views load with proper instructions and YouTube video links.

10. Troubleshooting:

If you encounter any issues, check the browser console for error messages. Common issues might include incorrect file paths, missing API keys, or network errors.

Ensure all external resources are correctly linked and accessible.

By following these setup instructions, you should have the AI-Powered Recipe Generator up and running smoothly on your local machine. Enjoy exploring new recipes and making the most of your culinary ingredients.

7. Functionalities and Implementation

a. Ingredient Input

Users can input an ingredient in the search box or select a category from the dropdown menu.
The application fetches data from TheMealDB API based on the input.

HTML:

Code:

```
<div class="meal-search-box">

  <select id="category-select">
```

```
        <option value="">Select a Category</option>
    </select>
    <input type="text" class="search-control" placeholder="Enter an ingredient" id="search-input">
    <button type="submit" class="search-btn btn" id="search-btn">
        <i class="fas fa-search"></i>
    </button>
</div>
```

Code explanation:

The provided HTML snippet creates a search interface for an AI recipe generator website. It includes three main components: a dropdown menu for selecting recipe categories, a text input field for entering ingredients, and a search button.

The `<div class="meal-search-box">` container wraps the entire search interface, providing structure and styling. Inside this container:

- The `<select id="category-select">` element represents a dropdown menu where users can select a recipe category (e.g., starter, vegan, vegetarian). The initial option, "Select a Category", prompts users to choose from available categories.

- The `<input type="text" class="search-control" placeholder="Enter an ingredient" id="search-input">` element is a text input field labeled "Enter an ingredient". Users can manually type in a main ingredient they have or want to use in their recipe search.

- The `<button type="submit" class="search-btn btn" id="search-btn">` button serves as the search trigger. It features a search icon (`<i class="fas fa-search"></i>`) from the FontAwesome library, indicating its function. Clicking this button initiates the search process based on the selected category or entered ingredient.

This search interface aims to enhance user experience by providing an intuitive way to explore recipes tailored to specific preferences or available ingredients. It integrates seamlessly into the frontend of the AI recipe generator website, facilitating user interaction and interaction with backend systems to retrieve and display relevant recipe data.

JavaScript:

Code:

```
const searchBtn = document.getElementById('search-btn');

const categorySelect = document.getElementById('category-select');

searchBtn.addEventListener('click', getMealList);

function getCategories() {

   fetch('https://www.themealdb.com/api/json/v1/1/list.php?c=list')

      .then(response => response.json())

      .then(data => {

         let html = '<option value="">Select a Category</option>';

         data.meals.forEach(category => {

            html += <option value="${category.strCategory}">${category.strCategory}</option>;

         });

         categorySelect.innerHTML = html;

      });

}

getCategories();
```

code explanation:

The provided JavaScript code snippet sets up functionality for an AI recipe generator website. It initializes an event listener on the search button (`search-btn`) to trigger a function (`getMealList`) when clicked. Additionally, it defines a function `getCategories()` that fetches recipe categories from an external API and populates them into a dropdown menu (`category-select`).

In detail, the `getCategories()` function utilizes the `fetch()` API to make a GET request to `https://www.themealdb.com/api/json/v1/1/list.php?c=list`, which retrieves a list of recipe categories in JSON format. Upon receiving a successful response, it converts the response data to JSON using `.json()`. Then, it dynamically generates HTML options for the dropdown menu based on the retrieved category data.

Each category retrieved (`category.strCategory`) is inserted into an HTML option element (`<option>`), with its `value` attribute set to the category name (`category.strCategory`). The resulting HTML is accumulated into the `html` variable, starting with a default option "Select a Category".

Finally, the generated HTML string (`html`) is assigned to the `innerHTML` property of the `categorySelect` element (`<select id="category-select">`). This dynamically updates the dropdown menu on the webpage, allowing users to select from a list of available recipe categories.

The `getCategories()` function is immediately invoked after its definition (`getCategories();`), ensuring that the dropdown menu is populated with categories as soon as the page loads. This setup prepares the frontend to handle user interactions effectively, facilitating streamlined recipe category selection and subsequent recipe searches based on user preferences.

b. Recipe List Display:

The application displays a list of matching recipes when the user searches for an ingredient or selects a category.

JavaScript:

Code:

```
const mealList = document.getElementById('meal');

function getMealList() {

    let selectedCategory = categorySelect.value;

    let searchInputTxt = document.getElementById('search-input').value.trim();

    let url = "";

    if (selectedCategory) {

        url = https://www.themealdb.com/api/json/v1/1/filter.php?c=${selectedCategory};

    } else if (searchInputTxt) {

        url = https://www.themealdb.com/api/json/v1/1/filter.php?i=${searchInputTxt};

    } else {

        alert('Please enter an ingredient or select a category');

        return;
```

```
}
fetch(url)
    .then(response => response.json())
    .then(data => {
        let html = "";
        if (data.meals) {
            data.meals.forEach(meal => {
                html += `
                    <div class="meal-item" data-id="${meal.idMeal}">
                        <div class="meal-img">
                            <img src="${meal.strMealThumb}" alt="food">
                        </div>
                        <div class="meal-name">
                            <h3>${meal.strMeal}</h3>
                            <a href="#" class="recipe-btn">Get Recipe</a>
                        </div>
                    </div>
                `;
            });
            mealList.classList.remove('notFound');
        } else {
            html = "Sorry, we didn't find any meal!";
            mealList.classList.add('notFound');
```

```
        }

        mealList.innerHTML = html;

    });

}
```

Code explanation:

The JavaScript code snippet provided is responsible for fetching and displaying meals from an external API based on user input from a category dropdown (`categorySelect`) or a text input field (`search-input`). Here's a detailed explanation:

The `getMealList()` function is triggered when the user clicks a search button or initiates a search action on the website. It begins by retrieving the selected category from the dropdown (`categorySelect.value`) and the trimmed value of the input field (`searchInputTxt`). Depending on whether a category or ingredient has been selected or entered, the function dynamically constructs a URL for fetching data from "The Meal DB" API (`https://www.themealdb.com/api/json/v1/1/`).

>URL Construction:

If a category is selected (`selectedCategory` has a value), the URL is structured to fetch meals based on that category (`url = `https://www.themealdb.com/api/json/v1/1/filter.php?c=${selectedCategory}`).

If an ingredient is entered (`searchInputTxt` is not empty), the URL is structured to fetch meals containing that ingredient (`url = `https://www.themealdb.com/api/json/v1/1/filter.php?i=${searchInputTxt}`).

If neither a category nor an ingredient is provided, an alert is displayed prompting the user to input either.

>Fetching Data:

The constructed URL is passed to the `fetch()` function, which makes an asynchronous request to the API.

Upon receiving a response, the data is converted to JSON format (`response.json()`).

>Rendering Results:

The function then dynamically generates HTML (`html`) based on the fetched meal data. For each meal (`meal` in `data.meals`), it creates a `div` element (`meal-item`) displaying the meal's image, name (`strMeal`), and a link (`recipe-btn`) to view the recipe details.

If meals are found (`data.meals` is not empty), the HTML for each meal is appended to `html`, and the class `notFound` is removed from `mealList` to display the meal items.

If no meals are found (`data.meals` is empty), a message "Sorry, we didn't find any meal!" is displayed, and the class `notFound` is added to `mealList` to indicate no results.

>Updating the UI:

Finally, the generated HTML (`html`) is assigned to the `innerHTML` of the `mealList` element (`<div id="meal">`). This updates the webpage with the fetched meal items or a relevant message based on the API response.

This implementation ensures that the AI recipe generator website dynamically updates and displays meal results based on user input, providing an interactive and responsive experience for users exploring recipes based on their preferences or available ingredients.

c. Recipe Selection and Details:

When a user clicks on a recipe, detailed instructions, images, and a link to a YouTube video are displayed.

JavaScript:

Code:

```
const mealDetailsContent = document.querySelector('.meal-details-content');

const recipeCloseBtn = document.getElementById('recipe-close-btn');

mealList.addEventListener('click', getMealRecipe);

function getMealRecipe(e) {
    e.preventDefault();
    if (e.target.classList.contains('recipe-btn')) {
        let mealItem = e.target.parentElement.parentElement;
        fetch(https://www.themealdb.com/api/json/v1/1/lookup.php?i=${mealItem.dataset.id})
            .then(response => response.json())
            .then(data => mealRecipeModal(data.meals));
    }
```

```
}
function mealRecipeModal(meal) {
    meal = meal[0];
    let html = `
        <h2 class="recipe-title">${meal.strMeal}</h2>
        <p class="recipe-category">${meal.strCategory}</p>
        <div class="recipe-instruct">
            <h3>Instructions:</h3>
            <p>${meal.strInstructions}</p>
        </div>
        <div class="recipe-meal-img">
            <img src="${meal.strMealThumb}" alt="">
        </div>
        <div class="recipe-link">
            <a href="${meal.strYoutube}" target="_blank">Watch Video</a>
        </div>
    `;
    mealDetailsContent.innerHTML = html;
    mealDetailsContent.parentElement.classList.add('showRecipe');
}
recipeCloseBtn.addEventListener('click', () => {
    mealDetailsContent.parentElement.classList.remove('showRecipe');
});
```

Code explanation:

The provided JavaScript code manages the display of meal recipes on an AI recipe generator website. It includes functionality to fetch and show detailed recipe information when a user clicks on a "Get Recipe" button associated with a meal.

The `getMealRecipe()` function is triggered by a click event on the `mealList` element, specifically targeting buttons with the class `recipe-btn`. When such a button is clicked, the function prevents the default action, identifies the specific meal item clicked (`mealItem`), and retrieves additional meal details from "The Meal DB" API using its unique identifier (`dataset.id`).

Upon receiving a response from the API, the `mealRecipeModal()` function is invoked. This function formats the fetched meal data (`meal`) into HTML, including the meal's title (`strMeal`), category (`strCategory`), cooking instructions (`strInstructions`), an image (`strMealThumb`), and a link to a cooking video on YouTube (`strYoutube`). This HTML content is then injected into the `mealDetailsContent` element, which is part of a modal window (`showRecipe` class) displayed on the webpage.

The `recipeCloseBtn` event listener allows users to close the recipe modal window by clicking on the close button (`recipe-close-btn`), which removes the `showRecipe` class from the modal's parent element, thereby hiding the modal and returning the user to the main interface.

Overall, this JavaScript functionality enhances user interaction by allowing them to explore detailed recipes and instructional videos associated with each meal displayed on the AI recipe generator website, promoting a seamless and engaging user experience.

8. Future Enhancements

The AI-Powered Recipe Generator is a robust application, but there's always room for enhancements and additional features to further enrich the user experience. Here are some potential future improvements that could be made:

1. User Authentication and Profile Management:

Implement a user authentication system that allows users to create accounts, log in, and manage their profiles.

Users could save their favorite recipes, create custom shopping lists, and track their cooking history.

2. Enhanced Search Functionality:

Improve the search functionality to allow for more complex queries, such as multiple ingredient searches, dietary restrictions (e.g., gluten-free, vegan), and cuisine types.

Implement advanced filters and sorting options to help users find recipes that best match their preferences.

3. Personalized Recommendations:

Utilize machine learning algorithms to provide personalized recipe recommendations based on users' search history, saved recipes, and ratings.

Develop a recommendation engine that suggests new recipes each time the user logs in.

4. Ingredient Inventory Management:

Add a feature for users to manage their ingredient inventory. Users can input the ingredients they have at home, and the application can suggest recipes based on their current inventory.

Include notifications for low-stock ingredients and suggestions for replenishing common items.

5. Meal Planning and Scheduling:

Integrate a meal planning feature that allows users to schedule their meals for the week or month.

Users can generate shopping lists based on their meal plans, ensuring they have all the ingredients needed for their recipes.

6. Social Sharing and Community Features:

Enable users to share their favorite recipes on social media platforms or within the application's community.

Develop a community section where users can post their own recipes, rate and review others' recipes, and engage with other cooking enthusiasts.

7. Voice Search and Commands:

Implement voice search functionality to allow users to search for recipes and navigate the application using voice commands.

Integrate with virtual assistants like Amazon Alexa or Google Assistant for hands-free operation.

8. Nutritional Information and Health Tracking:

Provide detailed nutritional information for each recipe, including calories, macronutrients, and micronutrients.

Allow users to track their dietary intake and set nutritional goals, helping them maintain a balanced and healthy diet.

9. Multilingual Support:

Expand the application to support multiple languages, making it accessible to a broader audience worldwide.

Implement language detection and translation features to enhance the user experience for non-English speakers.

10. Offline Mode:

Develop an offline mode that allows users to access saved recipes and shopping lists without an internet connection.

Ensure that users can still use essential features of the application while offline.

11. Interactive Cooking Mode:

Introduce an interactive cooking mode that guides users through the recipe step-by-step with visual and audio cues.

Include timers, progress indicators, and tips for each cooking step to make the process smoother and more enjoyable.

12. Integration with Grocery Delivery Services:

Partner with grocery delivery services to allow users to order ingredients directly from the application.

Provide users with the convenience of having ingredients delivered to their doorstep, streamlining the meal preparation process.

By implementing these future improvements, the AI-Powered Recipe Generator can evolve into a comprehensive and indispensable tool for home cooks, making meal planning, cooking, and grocery shopping easier and more enjoyable.

9. Algorithm for working of website

1. Fetch available categories from TheMealDB API and populate the dropdown.
2. Listen for user input (ingredient or category).
3. Fetch recipes based on user input.
4. Display a list of recipes.
5. Listen for recipe selection.
6. Fetch detailed recipe information.
7. Display recipe details, including instructions and a YouTube video link.
8. Allow user to close the recipe detail view.

10.Conclusion

The AI-Powered Recipe Generator is designed to simplify the process of finding and following recipes based on available ingredients. With a clean interface and interactive features, it aims to be a helpful tool for anyone looking to explore new meals and cooking techniques.

11. Reference

[1] Alayrac, J. B., Recasens, A., Schneider, R., Arandjelović, R., Ramapuram, J., De Fauw, J., ... & Zisserman, A. (2020). Self-supervised multimodal versatile networks. Advances in Neural Information Processing Systems, 33, 25-37.

[2] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PMLR.

[3] Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., ... & de Freitas, N. (2022). A generalist agent. arXiv preprint arXiv:2205.06175.

[4] Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., ... & Jitsev, J. (2022). Laion-5b: An open large-scale dataset for training next generation image-text models. arXiv preprint arXiv:2210.08402. https://laion.ai/blog/laion-5b/

[5] Kiros, R., Salakhutdinov, R., & Zemel, R. (2014, June). Multimodal neural language models. In International conference on machine learning (pp. 595-603). PMLR.

[6] Hossain, M. Z., Sohel, F., Shiratuddin, M. F., & Laga, H. (2019). A comprehensive survey of deep learning for image captioning. ACM Computing Surveys (CsUR), 51(6), 1-36.

[7] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... & Sutskever, I. (2021, July). Zeroshot text-to-image generation. In International Conference on Machine Learning (pp. 8821-8831). PMLR.

[8] Pahde, F., Puscas, M., Klein, T., & Nabi, M. (2021). Multimodal prototypical networks for few-shot learning. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 2644-2653).

[9] Goyal, P., Duval, Q., Seessel, I., Caron, M., Singh, M., Misra, I., ... & Bojanowski, P. (2022). Vision models are more robust and fair when pretrained on uncurated images without supervision. arXiv preprint arXiv:2202.08360.

[10] Fei, N., Lu, Z., Gao, Y., Yang, G., Huo, Y., Wen, J., ... & Wen, J. R. (2022). Towards artificial general intelligence via a multimodal foundation model. Nature Communications, 13(1), 3094.

[11] Todorova, M. (2020). " Narrow AI" in the Context of AI Implementation, Transformation and the End of Some Jobs. Nauchni trudove, (4), 15-25.

[12] CLIP-interrogator (2023), Google Colab notebook, https://colab.research.google.com/github/pharmapsychotic/clipinterrogator/blob/main/clip_interrogator.ipynb

[13] Witteveen, S., & Andrews, M. (2022). Investigating Prompt Engineering in Diffusion Models. arXivpreprint arXiv:2211.15462.

[14] Kim, C. (2019, March). A modular framework for collaborative multimodal annotation and visualization. In Proceedings of the 24th International Conference on Intelligent User Interfaces: Companion (pp. 165- 166).

[15] Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., ... & Petersen, S. (2016). Deepmind lab. arXiv preprint arXiv:1612.03801.

[16] Research paper – Autochef:Automated generation of cooking recipes ,Authors – Hajira Jabeen,Jonas weinz and Jens lehmann .Published under IEEE in the year of 2020

[17] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[18] D. B. T. Teisberg, "Recipe for disaster: A seq2seq model for recipe," 2018.

[19] .-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[20] S. N. R. Gona and H. Marellapudi, "Suggestion and invention of recipes using bi-directional lstms-based frameworks," *SN Applied Sciences*, vol. 3, no. 5, pp. 1–17, 2021.

[21] Z. Yu, H. Zang, and X. Wan, "Routing enforced generative model for recipe generation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 3797– 3806.

[22] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Advances in neural information processing systems*, vol. 30, 2017.

[23] B. P. Majumder, S. Li, J. Ni, and J. McAuley, "Generating personalized recipes from historical user preferences," *arXiv preprint arXiv:1909.00105*, 2019.

[24] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[25] B. Aljbawi, "Health-aware food planner: A personalized recipe generation approach based on gpt-2," 2020.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[28] M. Bien, M. Gilski, M. Maciejewska, and W. Taisner, "Cooking recipes´ generator utilizing a deep learning-based language model," Ph.D. dissertation, 02 2020.

[29] H. H. Lee, K. Shu, P. Achananuparp, P. K. Prasetyo, Y. Liu, E.-P. Lim, and L. R. Varshney, "Recipegpt: Generative pre-training based cooking recipe generation and evaluation system," in *Companion Proceedings of the Web Conference 2020*, 2020, pp. 181–184.

[30] J. Marin, A. Biswas, F. Ofli, N. Hynes, A. Salvador, Y. Aytar, I. Weber, and A. Torralba, "Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 187–203, 2019