### ****Recipe Finder Web App - Vanilla JavaScript Implementation****

#### ****Overview****

The Recipe Finder web app is a single-page application (SPA) built using **Vanilla JavaScript** (plain JavaScript without frameworks). It allows users to search for recipes by ingredient and view detailed information about selected recipes.

### ****Key JavaScript Features Used****

#### ****1. Fetching Data from an API****

The app fetches recipe data from the **Spoonacular API** using the fetch() method.

* The searchRecipes() function sends a request with a search query and displays the results.
* The showRecipeDetails() function fetches additional details when a recipe is clicked.

async function searchRecipes() {

const query = document.getElementById('search').value;

if (!query) return alert("Please enter an ingredient");

const apiKey = "20e8d773027b4b1b9cd8f47f2f39db49";

const url = `https://api.spoonacular.com/recipes/complexSearch?query=${query}&number=5&apiKey=${apiKey}`;

try {

const response = await fetch(url);

const data = await response.json();

displayRecipes(data.results);

} catch (error) {

alert("Failed to fetch recipes. Try again!");

}

}

#### ****2. Dynamically Displaying Recipes****

The displayRecipes() function dynamically creates HTML elements and appends them to the results container.

function displayRecipes(recipes) {

const resultsDiv = document.getElementById('results');

resultsDiv.innerHTML = "";

recipes.forEach(recipe => {

const recipeDiv = document.createElement("div");

recipeDiv.classList.add("recipe");

recipeDiv.innerHTML = `

<h3>${recipe.title}</h3>

<img src="${recipe.image}" alt="${recipe.title}" />

`;

recipeDiv.addEventListener("click", () => showRecipeDetails(recipe.id));

resultsDiv.appendChild(recipeDiv);

});

}

* The innerHTML method is used to insert recipe titles and images.
* addEventListener("click", ...) is used to attach a click event to each recipe to fetch and display its details.

#### ****3. Displaying Recipe Details Without Reloading****

When a recipe is clicked, showRecipeDetails() fetches more details and updates the #recipe-details section dynamically.

async function showRecipeDetails(recipeId) {

const apiKey = "20e8d773027b4b1b9cd8f47f2f39db49";

const url = `https://api.spoonacular.com/recipes/${recipeId}/information?apiKey=${apiKey}`;

try {

const response = await fetch(url);

const recipe = await response.json();

const detailsDiv = document.getElementById('recipe-details');

detailsDiv.innerHTML = `

<h2>${recipe.title}</h2>

<img src="${recipe.image}" alt="${recipe.title}" />

<p><strong>Ingredients:</strong></p>

<ul>${recipe.extendedIngredients.map(ing => `<li>${ing.original}</li>`).join('')}</ul>

<p><strong>Instructions:</strong></p>

<p>${recipe.instructions || "No instructions available."}</p>

`;

detailsDiv.style.display = "block";

} catch (error) {

alert("Failed to fetch recipe details. Try again!");

}

}

* This function fetches full details of the recipe, including ingredients and instructions.
* The .map() method is used to generate a list of ingredients.
* The innerHTML method updates the details section without reloading the page.

### ****How to Run the Web App****

1. **Open** index.html **in a browser** (no need for a server).
2. Enter an ingredient (e.g., "chicken") and click **Search** to fetch recipes.
3. Click on any recipe to see the details on the same page.

This project demonstrates the use of **API integration, DOM manipulation, and event handling** in Vanilla JavaScript.

****Screenshots:****



