

Recipe Finder App Guide

Let's break this down step by step so you can understand how to build your **Recipe Finder App** using JavaScript.

Step 1: Understanding the App Flow

1. The user enters an ingredient in an input field.
2. The app fetches recipe data from an external API based on the ingredient.
3. The app dynamically displays the recipe results on the page.

Step 2: Choosing a Recipe API

You can use an external API like [Spoonacular API](#) or [Edamam API](#). For this guide, I'll assume you're using **Spoonacular API**, which requires an API key.

- **API Endpoint:**

```
arduino
```

```
https://api.spoonacular.com/recipes/findByIngredients?ingredients=tomato,cheese&number=5&apiKey=
```

- Replace ``tomato,cheese`` with user input.
- ``number=5`` fetches 5 recipes.
- ``apiKey=YOUR_API_KEY`` is required to authenticate.

Step 3: Setting Up the HTML

We'll create a simple UI with:

- An input box for the user to enter ingredients.
- A button to fetch recipes.
- A section to display the recipes.

```
html
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Recipe Finder</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; }
    #recipes { display: flex; flex-wrap: wrap; justify-content: center; }
    .recipe { border: 1px solid #ddd; padding: 10px; margin: 10px; width: 200px; }
    .recipe img { width: 100%; border-radius: 10px; }
  </style>
</head>
<body>
  <h1>Recipe Finder</h1>
```

```
<input type="text" id="ingredient" placeholder="Enter ingredient (e.g., chicken)">
<button onclick="fetchRecipes()">Search</button>
<div id="recipes"></div>

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

Step 4: Fetching Data from API

- We'll use the **Fetch API** to send a request to Spoonacular.
- Convert the response to JSON.
- Extract the required details (recipe title, image, used ingredients).
- Display the results dynamically.

Key Concepts:

1. Event Handling

- When the user clicks the "Search" button, it triggers `fetchRecipes()`.

2. Fetch API Usage

- Sends a GET request to the API and retrieves recipe data.

3. JSON Parsing

- Extracts useful information from the API response.

4. DOM Manipulation

- Dynamically updates the HTML to display recipes.

Step 5: Writing JavaScript (script.js)

```
javascript

const apiKey = 'YOUR_API_KEY'; // Replace with your Spoonacular API key

async function fetchRecipes() {
  const ingredient = document.getElementById('ingredient').value.trim();
  if (ingredient === '') {
    alert('Please enter an ingredient.');
```

```

    recipesContainer.innerHTML = '<p>No recipes found. Try another ingredient.</p>';
    return;
}

recipes.forEach(recipe => {
    const recipeCard = document.createElement('div');
    recipeCard.classList.add('recipe');

    recipeCard.innerHTML = `
        <h3>${recipe.title}</h3>
        
        <p><strong>Used Ingredients:</strong></p>
        <ul>
            ${recipe.usedIngredients.map(ing => `<li>${ing.name}</li>`).join('')}
        </ul>
    `;

    recipesContainer.appendChild(recipeCard);
});
}

```

Step 6: Explanation of JavaScript

1. Fetching Data

javascript

```

const response = await fetch(url);
const data = await response.json();

```

- `fetch(url)`: Sends a request to the API.
- `await response.json()`: Converts the response into JSON format.

2. Handling Errors

javascript

```

try {
    // Fetching and displaying recipes
} catch (error) {
    console.error('Error fetching recipes:', error);
    alert('Failed to fetch recipes. Please try again.');
```

- **Why?** If there's an issue (e.g., network failure, invalid API key), the `catch` block handles it.

3. Displaying Data Dynamically

javascript

```

recipes.forEach(recipe => {
    const recipeCard = document.createElement('div');
    recipeCard.classList.add('recipe');

    recipeCard.innerHTML = `
        <h3>${recipe.title}</h3>
        
        <p><strong>Used Ingredients:</strong></p>
        <ul>
            ${recipe.usedIngredients.map(ing => `<li>${ing.name}</li>`).join('')}
        </ul>
    `;

```

```
`;  
    recipesContainer.appendChild(recipeCard);  
});
```

- `.forEach(recipe => { ... })``: Loops through each recipe.
- `document.createElement('div')``: Creates a new `div`` element.
- `recipe.usedIngredients.map(ing => `${ing.name}`).join('')``: Converts ingredients into an unordered list.

Step 7: Running the App

1. Get an API Key:

- Go to [Spoonacular](#) and create a free account.
- Copy your API key and replace `'YOUR_API_KEY'` in `script.js``.

2. Save Your Files:

- `index.html``
- `script.js``

3. Open `index.html`` in a Browser

- Enter an ingredient (e.g., "tomato").
- Click "Search" to see recipes.

Final Thoughts

• Enhancements:

- Add a **loading spinner** while fetching data.
- Allow users to click a recipe to view **detailed instructions**.
- Use **CSS Grid/Flexbox** to improve styling.

This project helps you **master API integration, Fetch API, JSON handling, and DOM manipulation** in JavaScript. 🚀 Let me know if you need any modifications or explanations!

Here is the complete code for your **Recipe Finder App** with all three files:

1. `index.html`` (HTML - UI Structure)

```
html  
  
<!DOCTYPE html>  
<html lang="en">  
<head>  
  <meta charset="UTF-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  <title>Recipe Finder</title>  
  <style>  
    body {  
      font-family: Arial, sans-serif;  
      text-align: center;  
      margin: 20px;  
      background-color: #f4f4f4;  
    }  
    h1 {  
      color: #333;  
    }  
  </style>  
</head>  
<body>  
  <h1>Recipe Finder</h1>  
  <div>  
    <input type="text" value="Search for a recipe by ingredient" />  
    <button type="button" value="Search" />  
  </div>  
</body>  
</html>
```

```

    }
    input {
      padding: 10px;
      width: 250px;
      margin-right: 10px;
      border: 1px solid #ccc;
      border-radius: 5px;
    }
    button {
      padding: 10px 15px;
      background-color: #28a745;
      color: white;
      border: none;
      border-radius: 5px;
      cursor: pointer;
    }
    button:hover {
      background-color: #218838;
    }
    #recipes {
      display: flex;
      flex-wrap: wrap;
      justify-content: center;
      margin-top: 20px;
    }
    .recipe {
      border: 1px solid #ddd;
      background: white;
      padding: 15px;
      margin: 10px;
      width: 200px;
      border-radius: 10px;
      box-shadow: 2px 2px 10px rgba(0,0,0,0.1);
    }
    .recipe img {
      width: 100%;
      border-radius: 10px;
    }
  }
</style>
</head>
<body>
  <h1>Recipe Finder</h1>
  <input type="text" id="ingredient" placeholder="Enter ingredient (e.g., chicken)">
  <button onclick="fetchRecipes()">Search</button>
  <div id="recipes"></div>

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

```

2. `script.js` (JavaScript - Fetch API & DOM Manipulation)

```

javascript

const apiKey = 'YOUR_API_KEY'; // Replace with your Spoonacular API key

async function fetchRecipes() {
  const ingredient = document.getElementById('ingredient').value.trim();
  if (ingredient === '') {
    alert('Please enter an ingredient.');
```

```

    return;
  }

  const url = `https://api.spoonacular.com/recipes/findByIngredients?ingredients=${ingredient}&nu
```

```

  try {
    const response = await fetch(url);
    if (!response.ok) throw new Error('Failed to fetch recipes');
    const data = await response.json();
    displayRecipes(data);
  }
}

```

```

    } catch (error) {
      console.error('Error:', error);
      alert('Error fetching recipes. Try again later.');
```

3. API Key (Where to Get It?)

- Go to [Spoonacular API](#).
- Sign up for a free account.
- Get your **API Key** and replace ``YOUR_API_KEY`` in ``script.js``.

How to Run the Project?

1. **Save the files** (``index.html``, ``script.js``) in the same folder.
2. **Open** ``index.html`` **in a browser**.
3. **Enter an ingredient** (e.g., "chicken").
4. **Click "Search"** and see the recipes.

This project will teach you:

- **Fetch API** (to get data from an external API).
- **JSON Handling** (extracting data).
- **DOM Manipulation** (dynamically updating the page).

Let me know if you have any questions! 🚀

A great **free alternative** to Spoonacular is **TheMealDB API**. It provides free access to a **large database of recipes** with images and ingredient lists.

API Alternative: TheMealDB

- **API URL (Search by Ingredient)**

```
bash
```

```
https://www.themealdb.com/api/json/v1/1/filter.php?i=chicken
```

- Replace `chicken` with the ingredient input.
- No API key required (**completely free**).
- Returns **meal name, image, and ID** (you can fetch full details using another API call).

Updated Code Using TheMealDB API

1. `index.html`

```
html

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Recipe Finder</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; margin: 20px; background-color: #f0f0f0; }
    h1 { color: #333; }
    input { padding: 10px; width: 250px; margin-right: 10px; border: 1px solid #ccc; border-radius: 5px; }
    button { padding: 10px 15px; background-color: #28a745; color: white; border: none; border-radius: 5px; }
    button:hover { background-color: #218838; }
    #recipes { display: flex; flex-wrap: wrap; justify-content: center; margin-top: 20px; }
    .recipe { border: 1px solid #ddd; background: white; padding: 15px; margin: 10px; width: 200px; }
    .recipe img { width: 100%; border-radius: 10px; }
  </style>
</head>
<body>
  <h1>Recipe Finder</h1>
  <input type="text" id="ingredient" placeholder="Enter ingredient (e.g., chicken)">
  <button onclick="fetchRecipes()">Search</button>
  <div id="recipes"></div>

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

2. `script.js`

```
javascript

async function fetchRecipes() {
  const ingredient = document.getElementById('ingredient').value.trim();
  if (ingredient === '') {
    alert('Please enter an ingredient.');
```

```
    return;
  }

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

  try {
    const response = await fetch(url);
    if (!response.ok) throw new Error('Failed to fetch recipes');
    const data = await response.json();
    displayRecipes(data.meals);
  } catch (error) {
    console.error('Error:', error);
    alert('Error fetching recipes. Try again later.');
```

```

    }
  }

  function displayRecipes(meals) {
    const recipesContainer = document.getElementById('recipes');
    recipesContainer.innerHTML = '';

    if (!meals) {
      recipesContainer.innerHTML = '<p>No recipes found. Try another ingredient.</p>';
      return;
    }

    meals.forEach(meal => {
      const recipeCard = document.createElement('div');
      recipeCard.classList.add('recipe');

      recipeCard.innerHTML = `
        <h3>${meal.strMeal}</h3>
        
        <p><a href="https://www.themealdb.com/meal/${meal.idMeal}" target="_blank">View Recipe<
      `;

      recipesContainer.appendChild(recipeCard);
    });
  }
}

```

Why Use TheMealDB?

- ✓ **100% Free** – No API key required.
- ✓ **Simple to Use** – Just fetch recipes by ingredient.
- ✓ **Reliable** – Large collection of meals.
- ✓ **Full Recipe Details Available** – You can get complete instructions using another API call.

This updated version will work **without requiring an API key** and **fetch recipes instantly**. 🚀 Let me know if you need improvements!

ChatGPT can make mistakes. Check important info.