**Assignment 7**: Asynchronous Programming

Task: Create a simple web application that fetches data from an external API asynchronously. Use fetch or other asynchronous programming techniques to handle the data.

**Below is an example of a simple web application that fetches data from the JSONPlaceholder API asynchronously using the fetch function. The application displays the fetched data on the webpage.**

**HTML (index.html):**

| **<!DOCTYPE html>**  **<html lang="en">**  **<head>**  **<meta charset="UTF-8">**  **<meta name="viewport" content="width=device-width, initial-scale=1.0">**  **<title>Asynchronous Programming Example</title>**  **<link rel="stylesheet" href="styles.css">**  **</head>**  **<body>**  **<div class="container">**  **<h1>Asynchronous Programming Example</h1>**  **<button id="fetchDataButton">Fetch Data</button>**  **<div id="output"></div>**  **</div>**  **<script src="script.js" defer></script>**  **</body>**  **</html>** |
| --- |

**CSS (styles.css):**

| **body {**  **font-family: Arial, sans-serif;**  **margin: 0;**  **padding: 0;**  **display: flex;**  **align-items: center;**  **justify-content: center;**  **height: 100vh;**  **background-color: #f5f5f5;**  **}**  **.container {**  **text-align: center;**  **}**  **button {**  **font-size: 16px;**  **padding: 10px 20px;**  **cursor: pointer;**  **background-color: #4caf50;**  **color: #fff;**  **border: none;**  **border-radius: 4px;**  **}**  **button:hover {**  **background-color: #45a049;**  **}**  **#output {**  **margin-top: 20px;**  **}** |
| --- |

**JavaScript (script.js):**

| **document.getElementById('fetchDataButton').addEventListener('click', fetchData);**  **async function fetchData() {**  **const apiUrl = 'https://jsonplaceholder.typicode.com/todos/1';**  **try {**  **const response = await fetch(apiUrl);**  **if (!response.ok) {**  **throw new Error(`HTTP error! Status: ${response.status}`);**  **}**  **const data = await response.json();**  **displayData(data);**  **} catch (error) {**  **console.error('Error fetching data:', error.message);**  **}**  **}**  **function displayData(data) {**  **const outputElement = document.getElementById('output');**  **outputElement.innerHTML = `**  **<p><strong>User ID:</strong> ${data.userId}</p>**  **<p><strong>Title:</strong> ${data.title}</p>**  **<p><strong>Completed:</strong> ${data.completed ? 'Yes' : 'No'}</p>**  **`;**  **}** |
| --- |

In this example:

The HTML file includes a button to trigger the data fetching and a div (output) to display the fetched data.

The CSS file styles the button and the output div.

The JavaScript file adds an event listener to the button. When the button is clicked, it calls the fetchData function using the async/await syntax.

The fetchData function uses the fetch function to make an asynchronous request to the JSONPlaceholder API. It checks the response status and throws an error if it's not successful. If the response is successful, it converts the response body to JSON and calls the displayData function to show the data on the webpage.

The displayData function updates the content of the output div with the fetched data.

To test this, open the index.html file in a web browser and click the "Fetch Data" button. The application will fetch data from the API and display it on the webpage.