Assignment: Asynchronous Programming Exercises

Objective:

Develop a series of JavaScript exercises focusing on different asynchronous programming techniques such as callbacks, promises, and async/await, while integrating API calls to display data.

Evaluation Criteria:

Part 1: Callbacks { 25 marks }

Task 1: Html and CSS (5 Marks)

- Create an HTML file named "callbacks.html" with the basic structure of an HTML document. done
- Also focus on Styling and Design. And no need to focus on responsiveness. done

Task 2: Callback Implementation (10 Marks)

- Include a button and a div element in the HTML file. done
- Implement JavaScript functionality that utilizes a callback function to simulate a delay of 5 seconds when the button is clicked. done
- Update the text in the div element to display “Callback executed after 5 seconds” after the delay. done

Task 3: Fetch Data from API (10 Marks)

- Modify the callback implementation to fetch data from the JSONPlaceholder API (e.g., posts endpoint (https://dummyjson.com/posts)). done
- Display the fetched data (e.g., post titles) in the div element after the callback is executed. done

Part 2: Promises { 25 marks }

Task 1: HTML and CSS (5 Marks)

- Create an HTML file named "promises.html" with the basic structure of an HTML document. done
- Also focus on Styling and Design. And no need to focus on responsiveness. done

Task 2: Promise Implementation (15 Marks)

- Include a button and a div element in the HTML file. done
- Implement JavaScript functionality that creates a Promise to fetch data from the JSONPlaceholder API (e.g., posts endpoint (https://dummyjson.com/posts)). done
- Display "Loading..." in the div element while the Promise is pending and update the text to show the fetched data once the Promise is resolved. done

Task 3: Error Handling (5 Marks)

- Also focus on error handling. done
- If the Promise takes longer than 5 seconds to resolve, reject it with a message like "Operation timed out." done
- Display the error message in the div if the Promise is rejected. done

Part 3: Async/Await { 25 marks }

Task 1: HTML and CSS (5 Marks)

- Create an HTML file named "async-await.html" with the basic structure of an HTML document. done
- Also focus on Styling and Design. And no need to focus on responsiveness. done

Task 2: Async/Await Implementation (10 Marks)

- Include a button and a div element in the HTML file. done
- Implement JavaScript functionality that uses async/await to fetch data from the JSONPlaceholder API (e.g., posts endpoint (https://dummyjson.com/posts)). done
- Display "Loading..." in the div element while the data is being fetched and update the text to show the fetched data once it is received. done

Task 3: Error Handling (10 Marks)

- Implement error handling in the async/await implementation. done
- Handle errors such as network issues or timeouts gracefully and display error messages in the div element. done

Assessment:

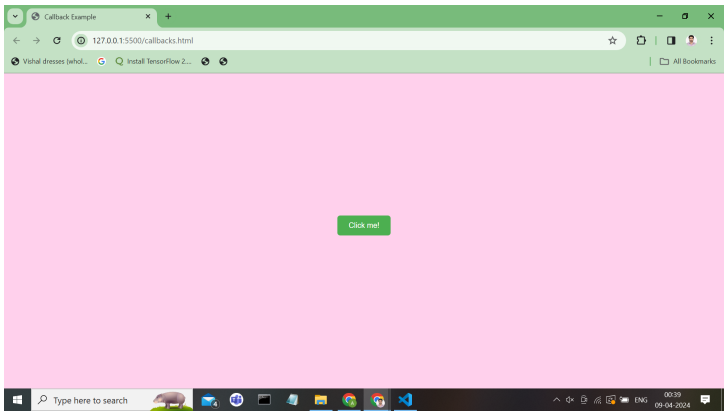
- Marks will be assigned for each task based on adherence to guidelines, implementation correctness, code quality, and overall understanding of asynchronous programming concepts. done

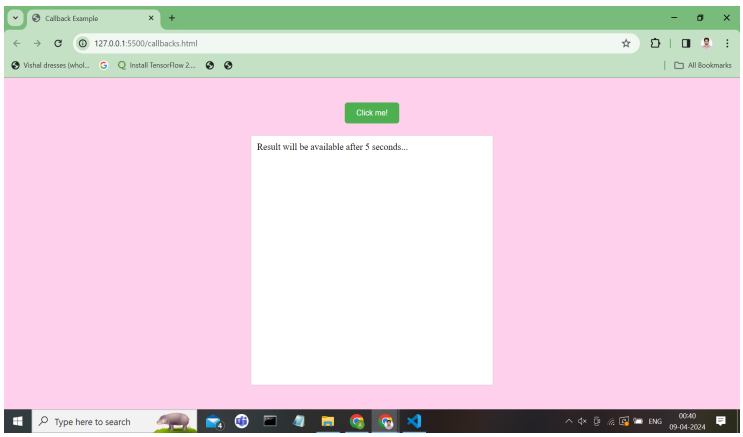
Submission Guidelines:

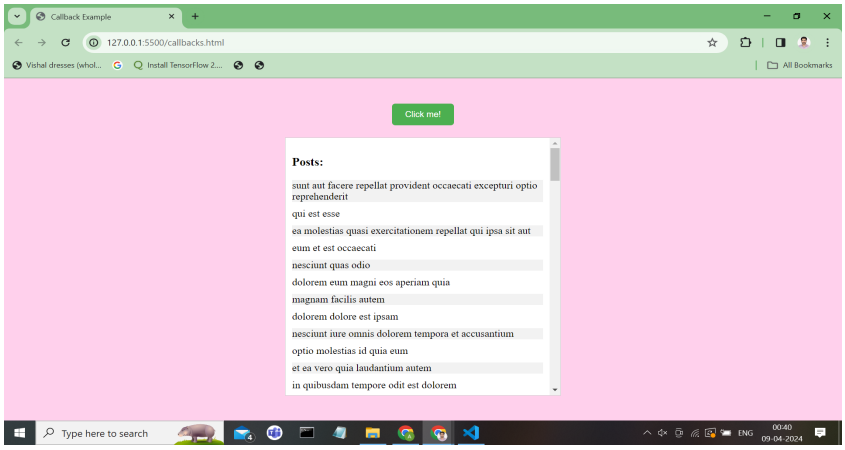
- Submit a zip file containing all HTML files, CSS file, and JavaScript file.
- Ensure code readability and organization, including appropriate comments for complex sections. done

Sample Image:

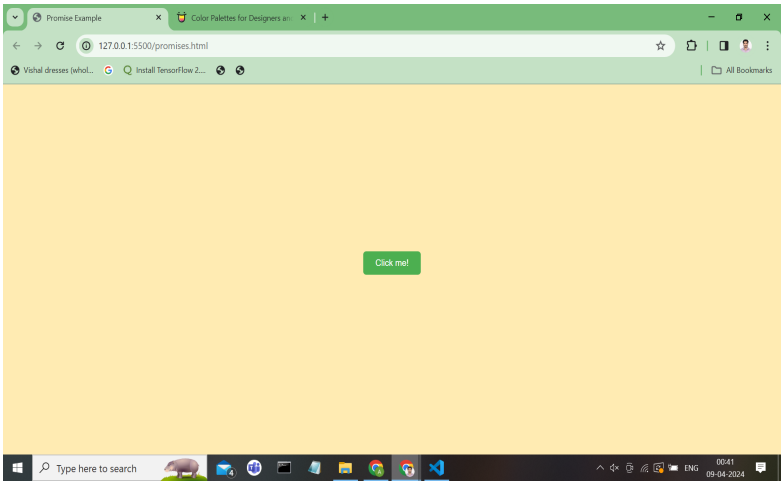
For Callbacks:

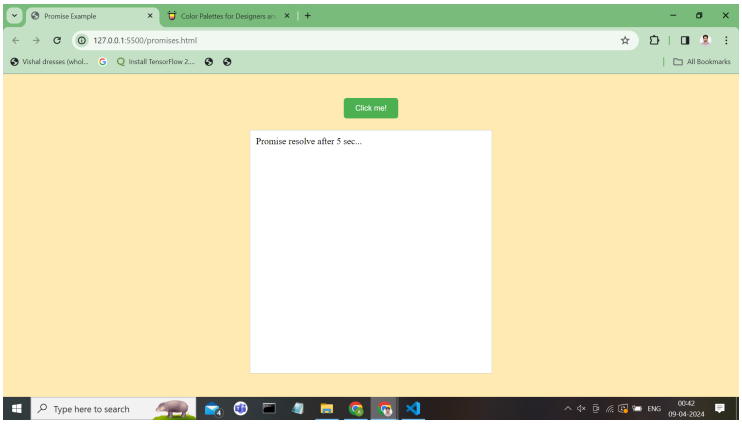


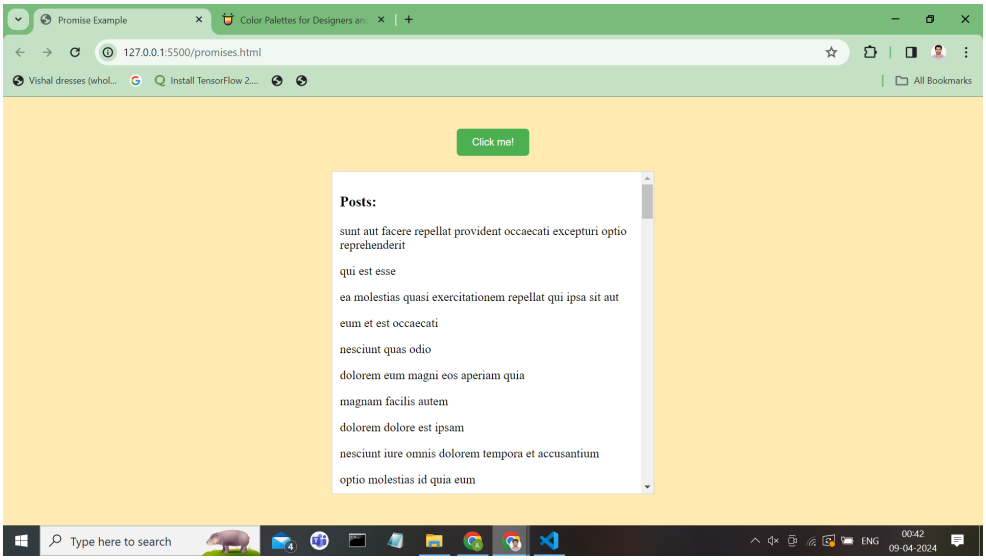




For Promise :







For Async Await: