Gmail Bikash Adhikari <virtual.bikash@gmail.com>

Task 1.1 : Simulating Asynchronous Behavior Create a...

1 message

Bikash <bikash.caim@gmail.com> Sat, Mar 8, 2025 at 6:24 PM
To: virtual.bikash@gmail.com

JavaScript

```javascript
function simulateAsyncTask() {
  console.log("Task started");
  setTimeout(() => {
    console.log("Task finished");
  }, 2000); // 2000 milliseconds = 2 seconds
}

// Example usage:
simulateAsyncTask();
```

Explanation:

1. `simulateAsyncTask()` **Function:**
   - This function is designed to mimic an asynchronous task.
   - It immediately logs "Task started" to the console.
   - It then uses `setTimeout()` to schedule a function to be executed after a delay.
2. `setTimeout()`**:**
   - `setTimeout()` is a built-in JavaScript function that allows you to delay the execution of a function.
   - It takes two arguments:
     - The first argument is the function to be executed.
     - The second argument is the delay in milliseconds.
   - In this case, the arrow function `() => { console.log("Task finished"); }` is set to execute after 2000 milliseconds (2 seconds).
3. **Asynchronous Behavior:**
   - The key to asynchronous behavior is that the program doesn't wait for the `setTimeout()` delay to finish before continuing.
   - The "Task started" message is logged immediately, and then the program moves on.
   - Two seconds later, the `setTimeout()` callback function is executed, and "Task finished" is logged.