4. Edutech: Student Records System

Objective: Manage student profiles, grades, and courses using callback functions, error handling, async/await, and ES6 features.

Part 1: Basic Features

Instructions:

1. **Create a class Student** with properties `name`, `age`, `grade`, and `courses`.
2. **Implement methods to:**
    - Update the student's grade (with error handling).
    - Add a new course (with callback function).
    - Fetch student data from a simulated API (using async/await).
    - Save updated student data to a simulated API (using async/await).
3. **Use callback functions** to display updated grade and courses after fetching and updating data.
4. **Add error handling** to manage simulated API request failures.

Part 2: Advanced Features

Objective: Enhance the student records system using advanced ES6 features, including modules, EventEmitter, and streams.

Instructions:

1. **Organize code using ES6 modules**:
    - Split the code into separate modules for Student, API calls, Logger, and Streams.
2. **Add an event-driven logging system** using EventEmitter:
    - Log significant actions such as fetching data, updating grade, and adding courses.
3. **Implement stream operations**:
    - Create readable, writable, duplex, and transform streams to handle student data.
    - Use `pipe()` for efficient data flow.
4. **Enhance error handling** to manage stream operations and API request failures.