Day 17 – Node.js Core Modules

Challenge 4: File System (fs) Module

User Story:
As a content manager, I want to store user input into a file and read it back for confirmation.

Problem Statement:
Use Node's **fs module** to:

- Write user input (feedback.txt)
- Then read and print the contents on the console.

Expected Outcome:
Input: "Node.js is awesome!"
Output:

Data written successfully.

Reading file...

Node.js is awesome!

Bonus:
Use fs.promises instead of callbacks.

Self-Evaluation Metrics:

Metric	Target
File created and written successfully	
Read content printed on console	
Used fs.promises or async version	

Challenge 5: HTTP Module

User Story:
As a backend developer, I want to serve a basic web page using Node.js without Express.

Problem Statement:
Create a simple HTTP server using Node's **http module** that:

- Serves "Hello from Node.js Server" at / route.
- Serves "About Page" at /about.

Expected Outcome:
Visit http://localhost:3000 → Displays "Hello from Node.js Server".

Bonus:
Serve static HTML file content instead of plain text.

Self-Evaluation Metrics:

Metric	Target
HTTP server created	
Handled multiple routes	
Properly closed server using ctrl + c	

Challenge 6: Events Module

User Story:
As a developer, I want to simulate event-driven behavior (like a notification system).

Problem Statement:
Use **EventEmitter** to create custom events:

- userLoggedIn
- userLoggedOut

  Trigger them with messages logged on the console.

Expected Outcome:

User John logged in.

User John logged out.

Bonus:

Emit a custom event sessionExpired after 5 seconds.

Self-Evaluation Metrics:

Metric	Target
Used EventEmitter correctly	
Registered event listeners	
Emitted events dynamically	