

Day 17 — Node.js Core Modules Project Report

Challenge 4 — File System (fs)

This challenge involved writing user-provided CLI input into a file named feedback.txt using fs.promises. After writing, the program reads the same file and prints the content back to the terminal.

Approach: The solution uses async/await, process.argv for CLI input, and writeFile/readFile methods to perform file operations efficiently and cleanly.

Output Screenshot:

The screenshot shows a Windows desktop environment with Visual Studio Code open. The code editor displays a file named 'challenge4_fs.js' which contains the following code:

```
const fs = require('fs').promises;
const path = require('path');

(async () => {
    try {
        // Take CLI input (3rd argument)
        const userInput = process.argv[2];

        if (!userInput) {
            console.log("Please pass input text. Example:");
            console.log("node challenge4_fs.js \"Your feedback here\"");
            return;
        }

        const filePath = path.join(__dirname, 'feedback.txt');

        // Write input from user
        await fs.writeFile(filePath, userInput, 'utf8');
        console.log('Data written successfully.');
    }
})()
```

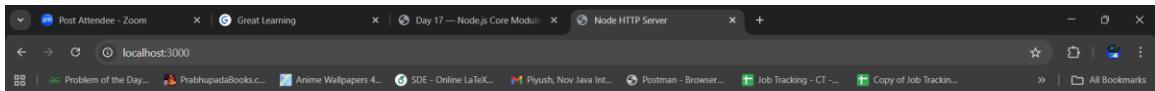
The terminal tab shows the command 'node challenge4_fs.js "Node.js is awesome!"' being run, and the output indicates the data was written successfully to 'feedback.txt'. The status bar at the bottom right shows the date and time as 14-11-2025, 11:20.

Challenge 5 — HTTP Module

This challenge required creating a basic Node.js HTTP server without Express, supporting routes '/', '/about', and returning appropriate responses.

Approach: The solution uses the built-in http module, route matching, and also implements a bonus feature to serve a static index.html file if present.

Home Page Output Screenshot:

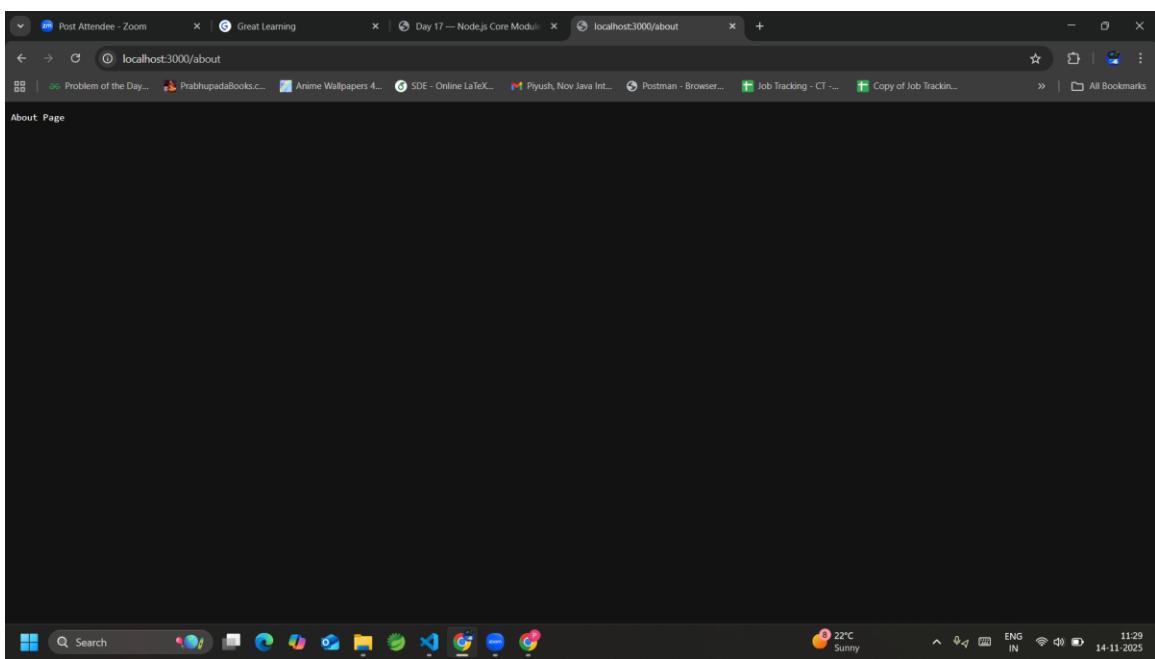


Hello from Node.js Server (static HTML)

This index.html was served as static content.



About Page Output Screenshot:



Challenge 6 — Events Module (EventEmitter)

This challenge involved simulating user activity by emitting events such as `userLoggedIn`, `userLoggedOut`, and a bonus `sessionExpired` event using Node's `EventEmitter`.

Approach: A custom notifier class extends `EventEmitter`, listeners are registered for all events, and timed emits simulate realistic user session behavior.

Output Screenshot:

File Edit Selection View Go Run ... ← → Q Day17_NodeJs_Core_Modules

EXPLORER JS challenge4_fs.js U JS challenge5_http.js U JS challenge6_events.js U X index.html U feedback.txt U

DAY17.NODEJS.CORE_MO... JS challenge6_events.js > ...

challenge4_fs.js U challenge5_http.js U challenge6_events.js U index.html U package.json U

7 // Register listeners
8 notifier.on('userLoggedIn', (username) => {
9 | console.log(`User \${username} logged in.`);
10 });
11
12 notifier.on('userLoggedOut', (username) => {
13 | console.log(`User \${username} logged out.`);
14 });
15
16 // Bonus: sessionExpired
17 notifier.on('sessionExpired', (username) => {
18 | console.log(`Session for \${username} expired.`);
19 });
20
21 // Simulate dynamic emits
22 function simulateUserActivity(username) {
23 | // user logs in
24 | notifier.emit('userLoggedIn', username);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\wsWipro\Wipro-MERN-FY26-Practice-Assignments\Day17_NodeJs_Core_Modules> node challenge6_events.js
User John logged in.
User John logged out.
Session for John expired.

PS D:\wsWipro\Wipro-MERN-FY26-Practice-Assignments\Day17_NodeJs_Core_Modules>

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF () JavaScript Go Live

Trending videos Stranger Things... ENG IN 11:39 14:11:2025