

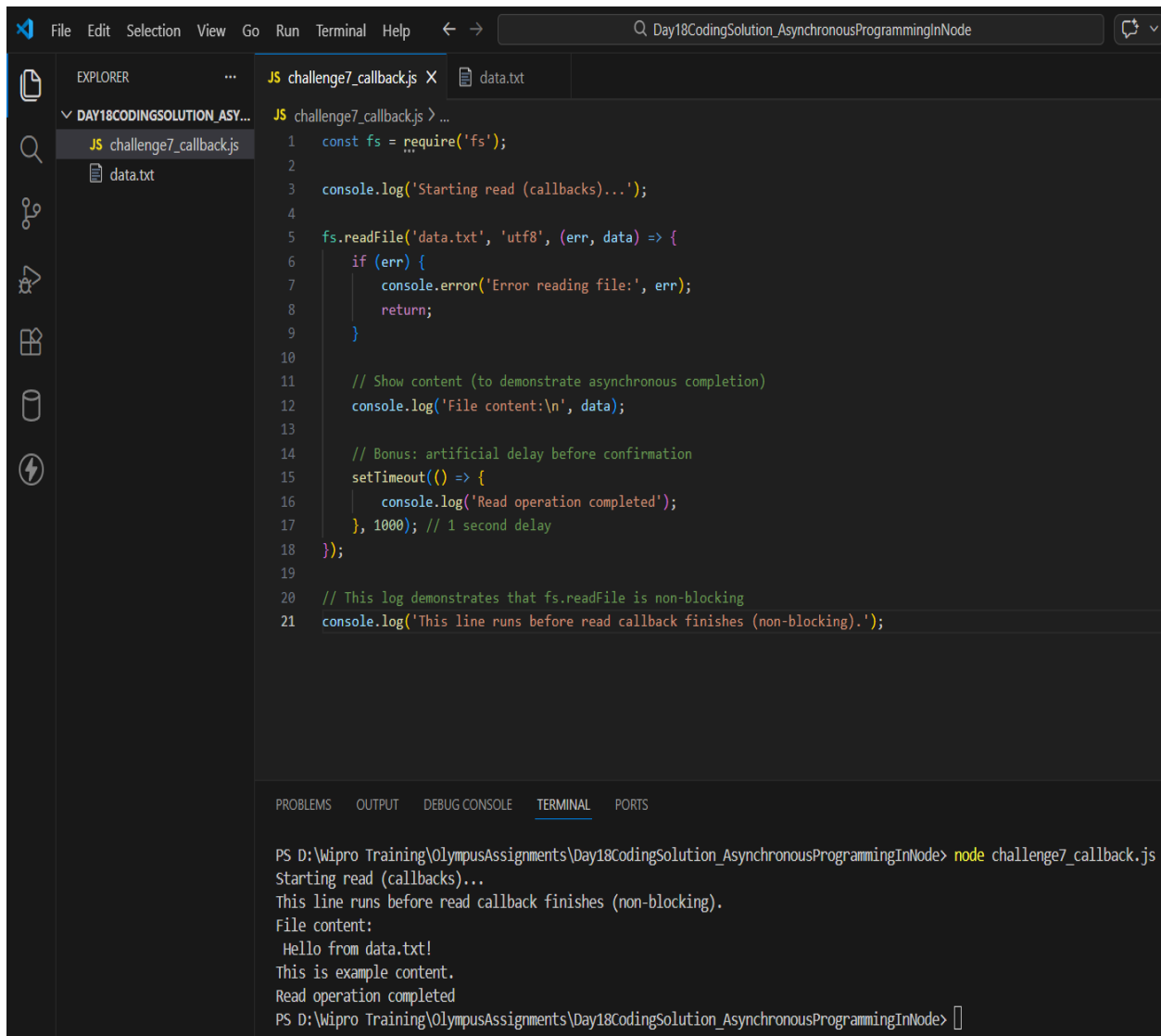
## Day 18 - Asynchronous Programming in Node.js

### Challenge 7: Callbacks

#### User Story

As a developer, I want to read a file and then display a confirmation message once it's done — using callbacks.

#### Code Snippet and Output



The screenshot shows a Visual Studio Code editor with a file named `challenge7_callback.js` open. The file contains the following JavaScript code:

```
1 const fs = require('fs');
2
3 console.log('Starting read (callbacks)...');
4
5 fs.readFile('data.txt', 'utf8', (err, data) => {
6   if (err) {
7     console.error('Error reading file:', err);
8     return;
9   }
10
11   // Show content (to demonstrate asynchronous completion)
12   console.log('File content:\n', data);
13
14   // Bonus: artificial delay before confirmation
15   setTimeout(() => {
16     console.log('Read operation completed');
17   }, 1000); // 1 second delay
18 });
19
20 // This log demonstrates that fs.readFile is non-blocking
21 console.log('This line runs before read callback finishes (non-blocking).');
```

The terminal output shows the execution of the script:

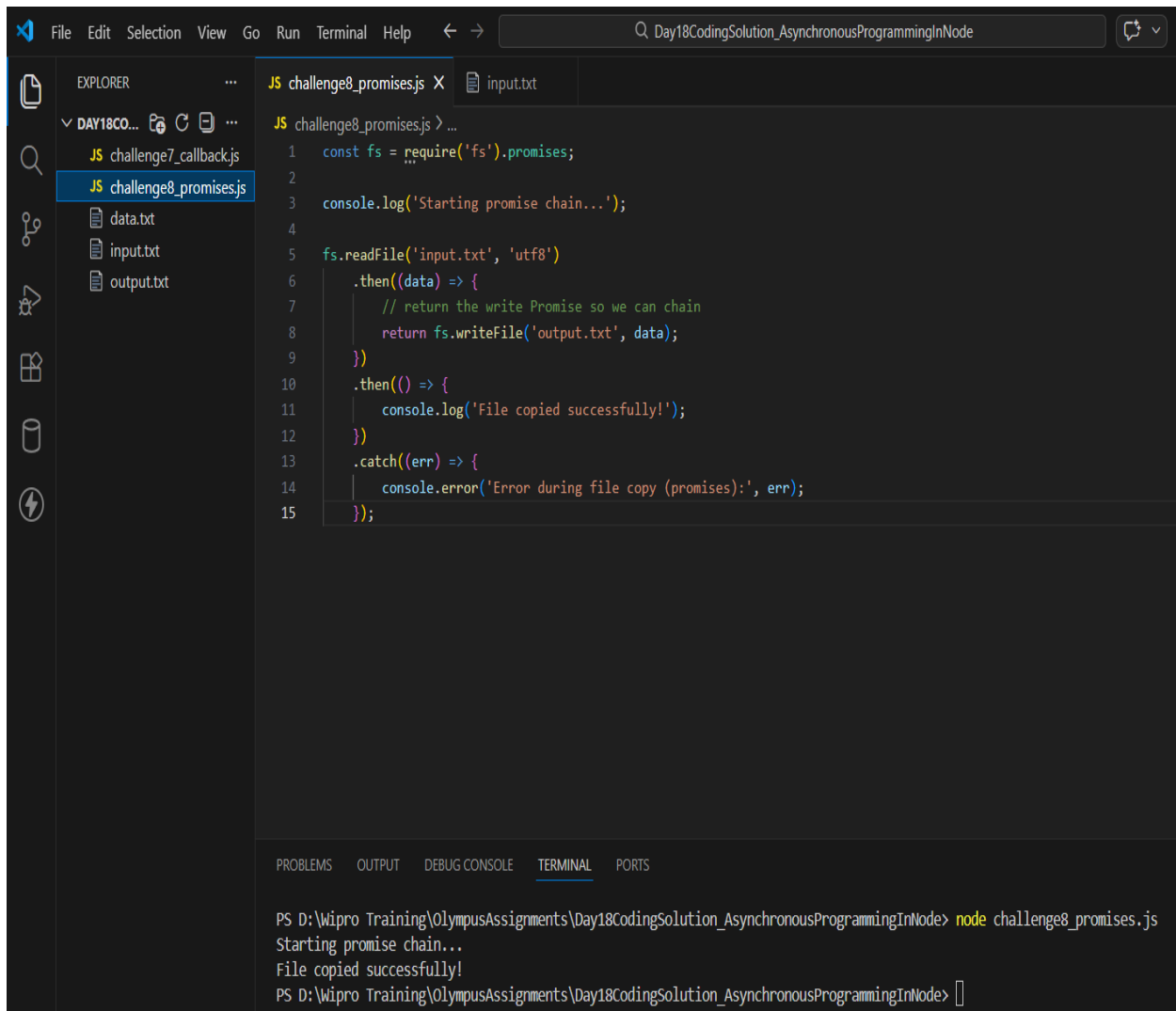
```
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode> node challenge7_callback.js
Starting read (callbacks)...
This line runs before read callback finishes (non-blocking).
File content:
Hello from data.txt!
This is example content.
Read operation completed
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode>
```

## Challenge 8: Promises

### User Story

As a developer, I want to chain multiple async operations (read file → write to another file) using Promises.

### Code Snippet and Output



The screenshot shows a Visual Studio Code editor window with the following components:

- Explorer Panel:** Shows a file tree with `challenge7_callback.js` and `challenge8_promises.js` selected. Below them are `data.txt`, `input.txt`, and `output.txt`.
- Editor Panel:** Displays the content of `challenge8_promises.js`. The code is as follows:

```
1 const fs = require('fs').promises;
2
3 console.log('Starting promise chain...');
4
5 fs.readFile('input.txt', 'utf8')
6   .then((data) => {
7     // return the write Promise so we can chain
8     return fs.writeFile('output.txt', data);
9   })
10  .then(() => {
11    console.log('File copied successfully!');
12  })
13  .catch((err) => {
14    console.error('Error during file copy (promises):', err);
15  });
```
- Terminal Panel:** Shows the command `node challenge8_promises.js` being executed, with the following output:

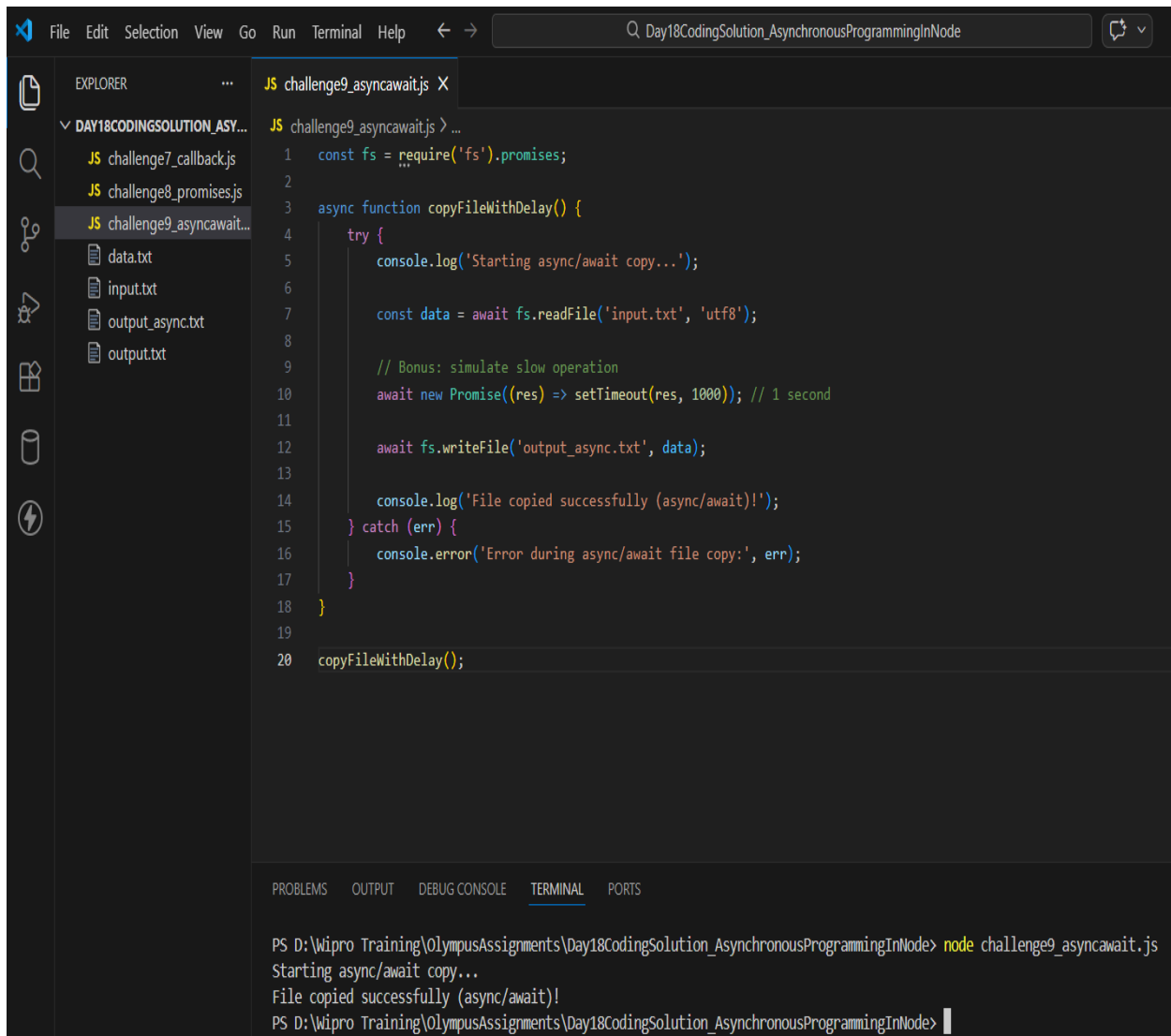
```
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode> node challenge8_promises.js
Starting promise chain...
File copied successfully!
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode>
```

## Challenge 9: Async/Await

### User Story

As a developer, I want cleaner syntax for asynchronous operations using modern JavaScript.

### Code Snippet and Output



The screenshot shows a Visual Studio Code editor window with a file named `challenge9_asyncawait.js` open. The file contains the following JavaScript code:

```
1  const fs = require('fs').promises;
2
3  async function copyFileWithDelay() {
4      try {
5          console.log('Starting async/await copy...');
6
7          const data = await fs.readFile('input.txt', 'utf8');
8
9          // Bonus: simulate slow operation
10         await new Promise((res) => setTimeout(res, 1000)); // 1 second
11
12         await fs.writeFile('output_async.txt', data);
13
14         console.log('File copied successfully (async/await)!');
15     } catch (err) {
16         console.error('Error during async/await file copy:', err);
17     }
18 }
19
20 copyFileWithDelay();
```

The terminal output at the bottom of the editor shows the command `node challenge9_asyncawait.js` being executed, with the following output:

```
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode> node challenge9_asyncawait.js
Starting async/await copy...
File copied successfully (async/await)!
PS D:\Wipro Training\OlympusAssignments\Day18CodingSolution_AsynchronousProgrammingInNode>
```