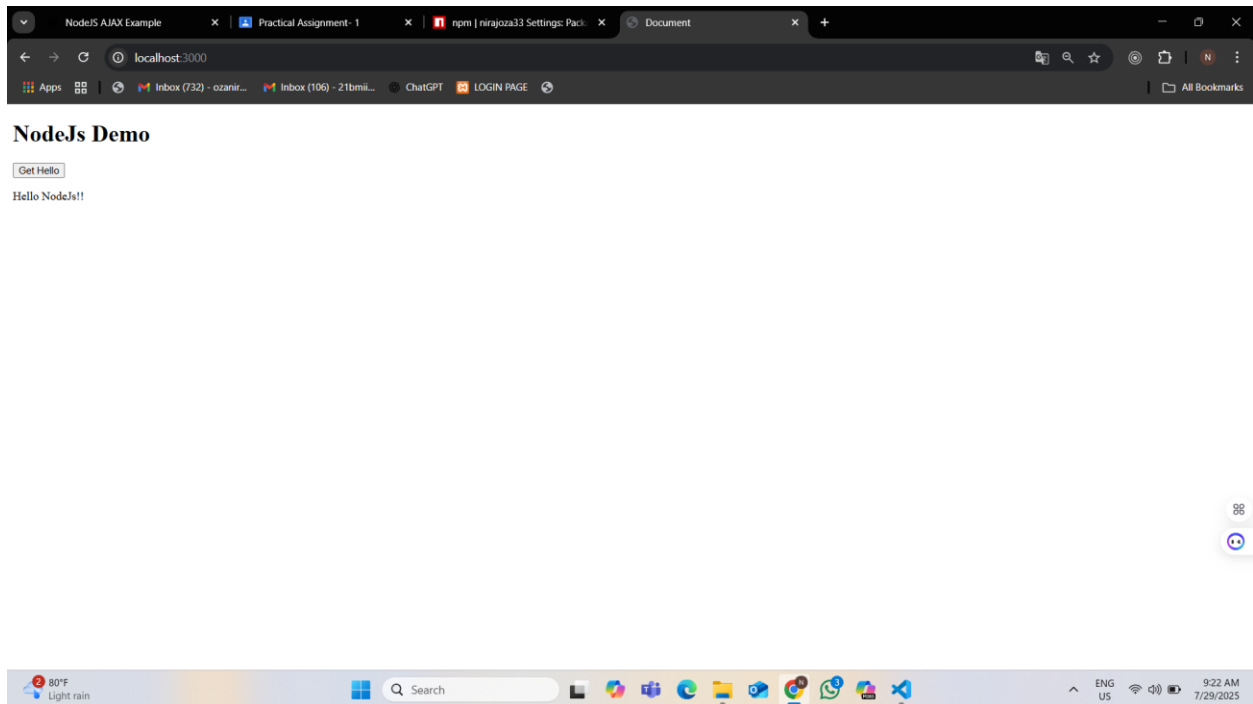


1. Develop nodejs application with following requirements:

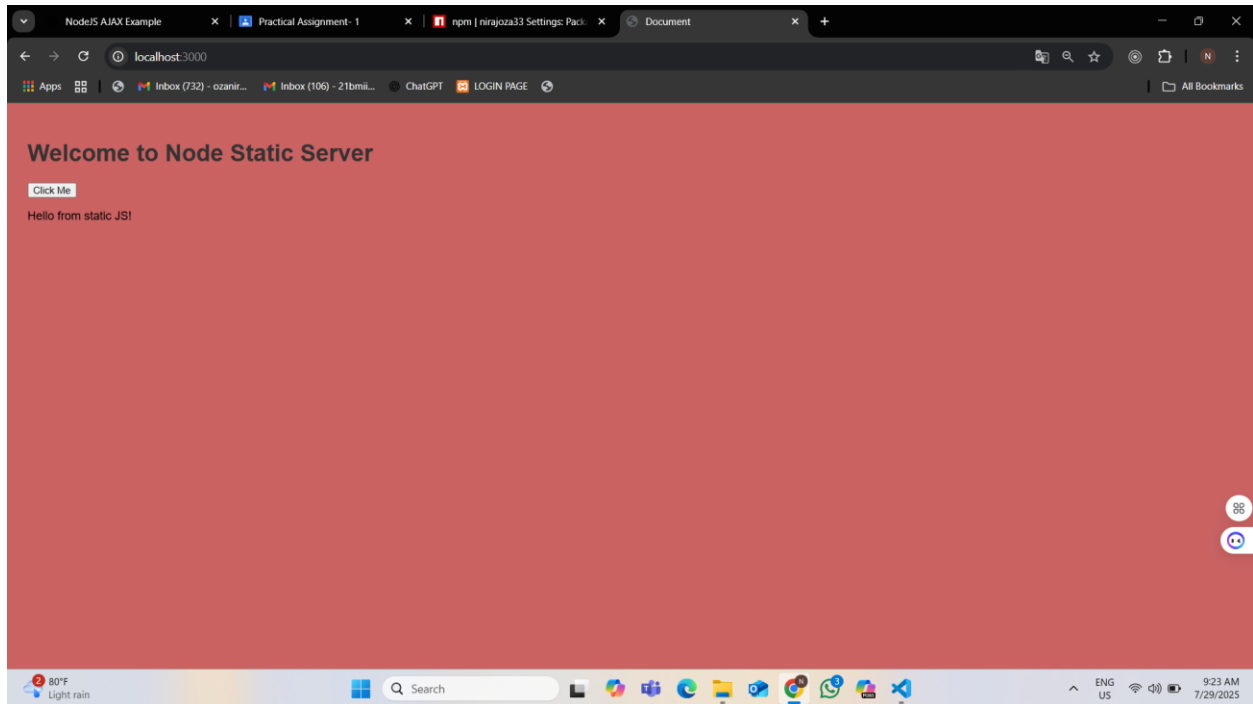
- Develop a route `"/gethello"` with GET method. It displays `"Hello NodeJS!!"` as response.
- Make an HTML page and display.
- Call `"/gethello"` route from HTML page using AJAX call. (Any frontend AJAX call API can be used.)

ScreenShot



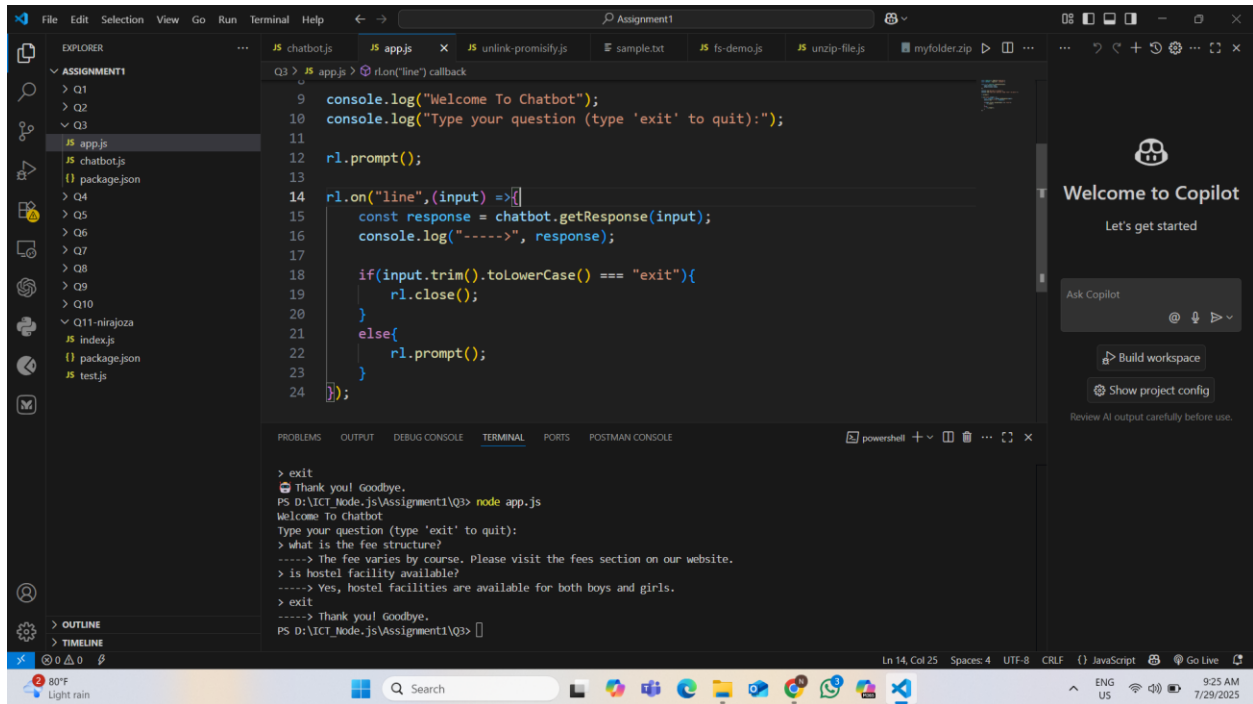
2. Develop a web server which serves static resources.

ScreenShot



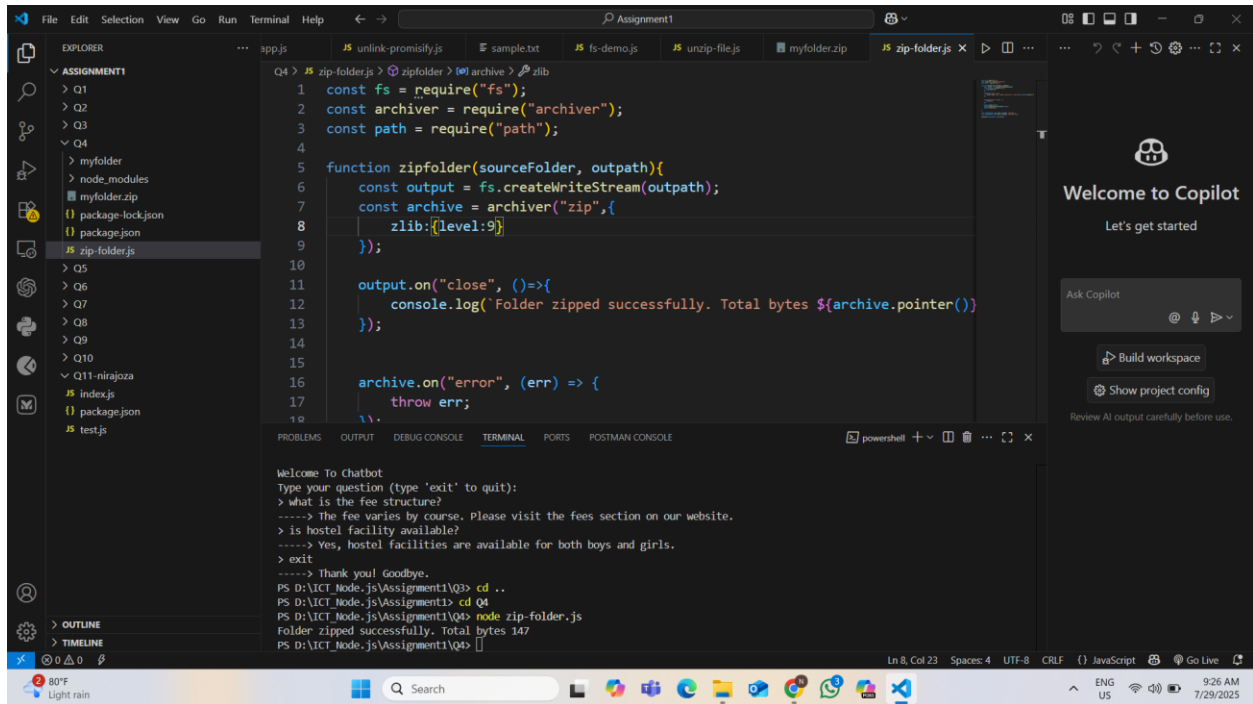
3. Develop a module for domain specific chatbot and use it in a command line application.

ScreenShot



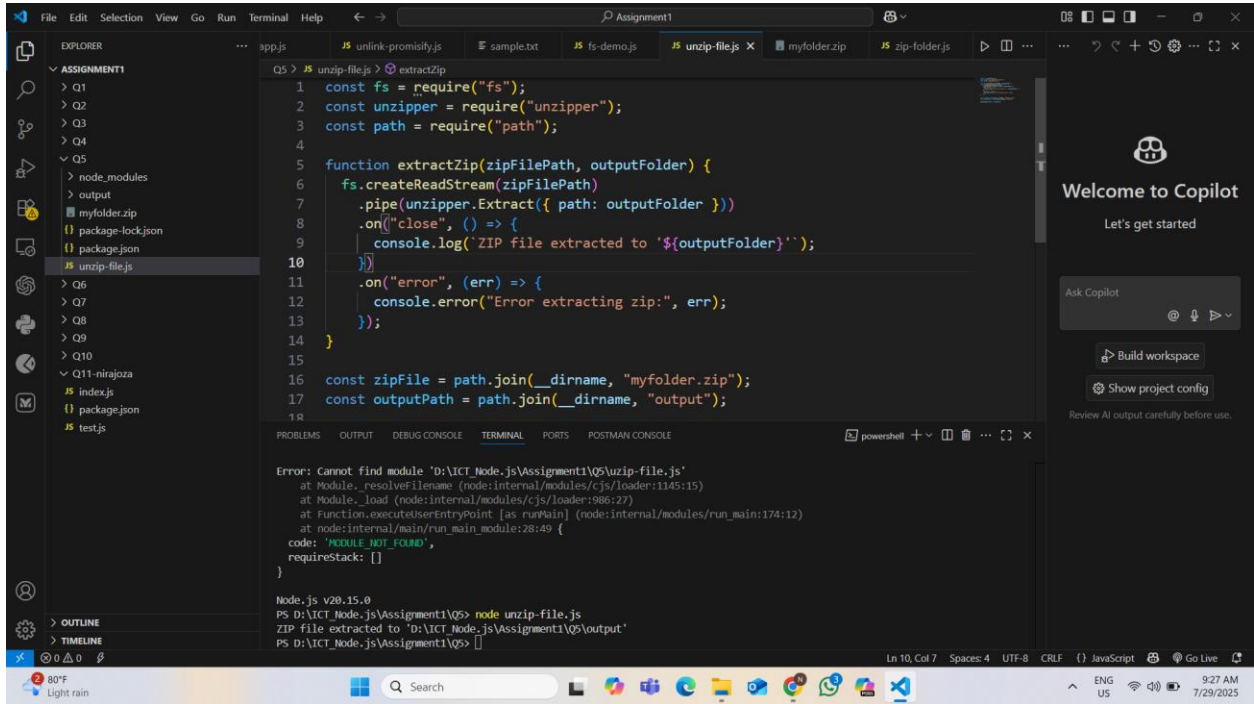
4. Write a program to create a compressed zip file for a folder.

ScreenShot



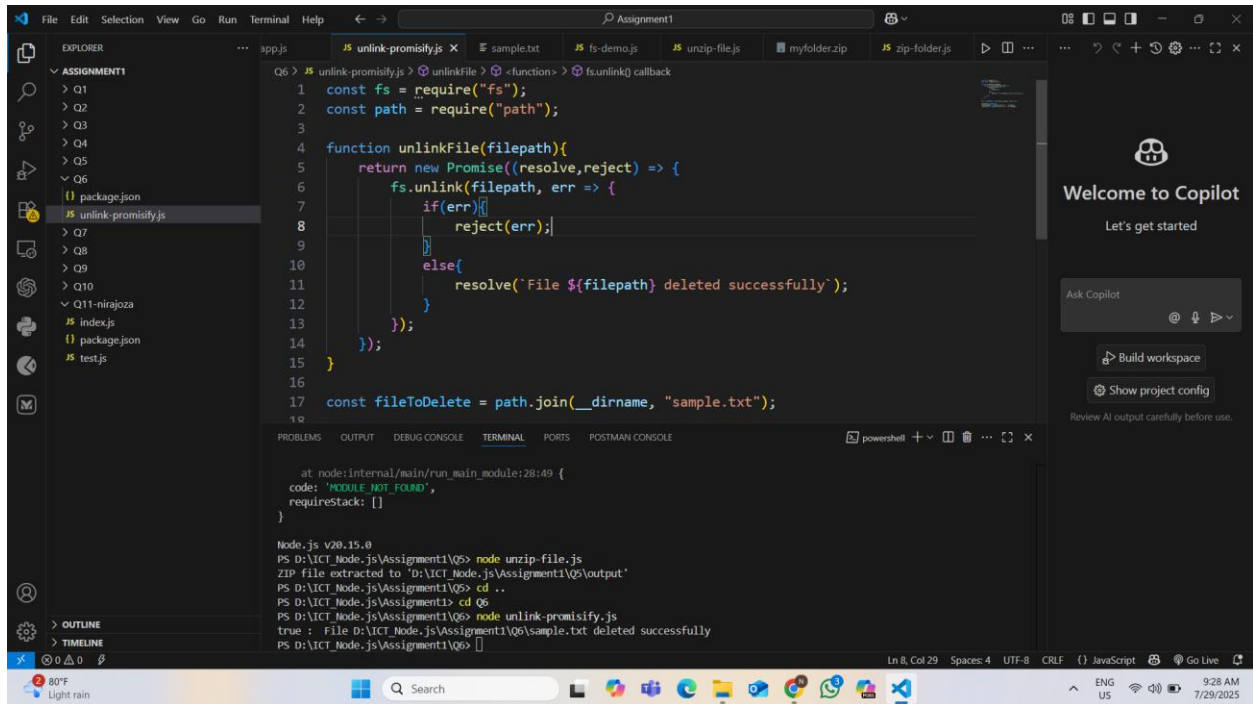
5. Write a program to extract a zip file.

ScreenShot



6. Write a program to promisify fs.unlink function and call it.

ScreenShot



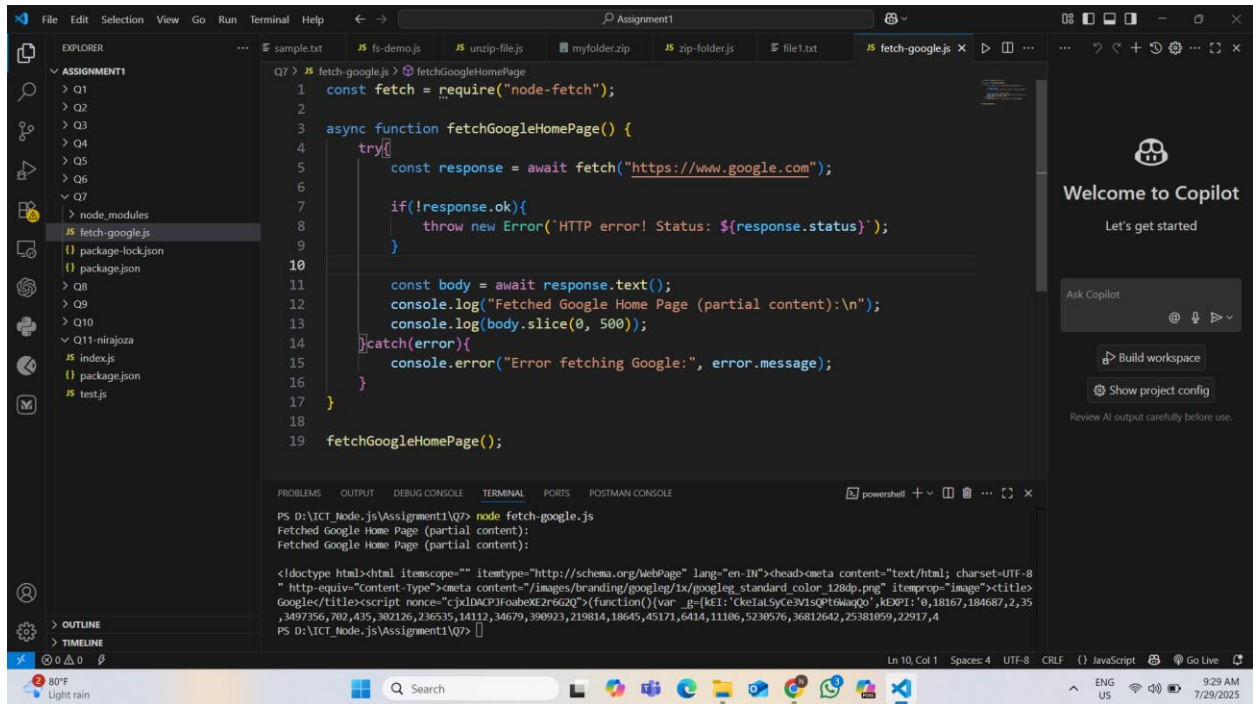
```
1 const fs = require("fs");
2 const path = require("path");
3
4 function unlinkFile(filepath){
5     return new Promise((resolve,reject) => {
6         fs.unlink(filepath, err => {
7             if(err){
8                 reject(err);
9             }
10            else{
11                resolve(`File ${filepath} deleted successfully`);
12            }
13        });
14    });
15 }
16
17 const fileToDelete = path.join(__dirname, "sample.txt");
```

```
at node:internal/main/run_main_module:28:49 {
  code: 'MODULE_NOT_FOUND',
  requireStack: []
}

Node.js v20.15.0
PS D:\ICT_Node.js\Assignment1\Q5> node unzip-file.js
ZIP file extracted to 'D:\ICT_Node.js\Assignment1\Q5\output'
PS D:\ICT_Node.js\Assignment1\Q5> cd ..
PS D:\ICT_Node.js\Assignment1> cd Q6
PS D:\ICT_Node.js\Assignment1\Q6> node unlink-promisify.js
true : File D:\ICT_Node.js\Assignment1\Q6\sample.txt deleted successfully
PS D:\ICT_Node.js\Assignment1\Q6>
```

7. Fetch data of google page using node-fetch using async-await model.

ScreenShot



The screenshot shows a Visual Studio Code editor window with a project named "Assignment1". The Explorer sidebar on the left shows a file tree with folders Q1 through Q11, and files like fetch-google.js, package-lock.json, package.json, index.js, and test.js. The main editor area displays the code for "fetch-google.js":

```
1 const fetch = require("node-fetch");
2
3 async function fetchGoogleHomePage() {
4     try{
5         const response = await fetch("https://www.google.com");
6
7         if(!response.ok){
8             throw new Error(`HTTP error! Status: ${response.status}`);
9         }
10
11         const body = await response.text();
12         console.log("Fetched Google Home Page (partial content):\n");
13         console.log(body.slice(0, 500));
14     }catch(error){
15         console.error("Error fetching Google:", error.message);
16     }
17 }
18
19 fetchGoogleHomePage();
```

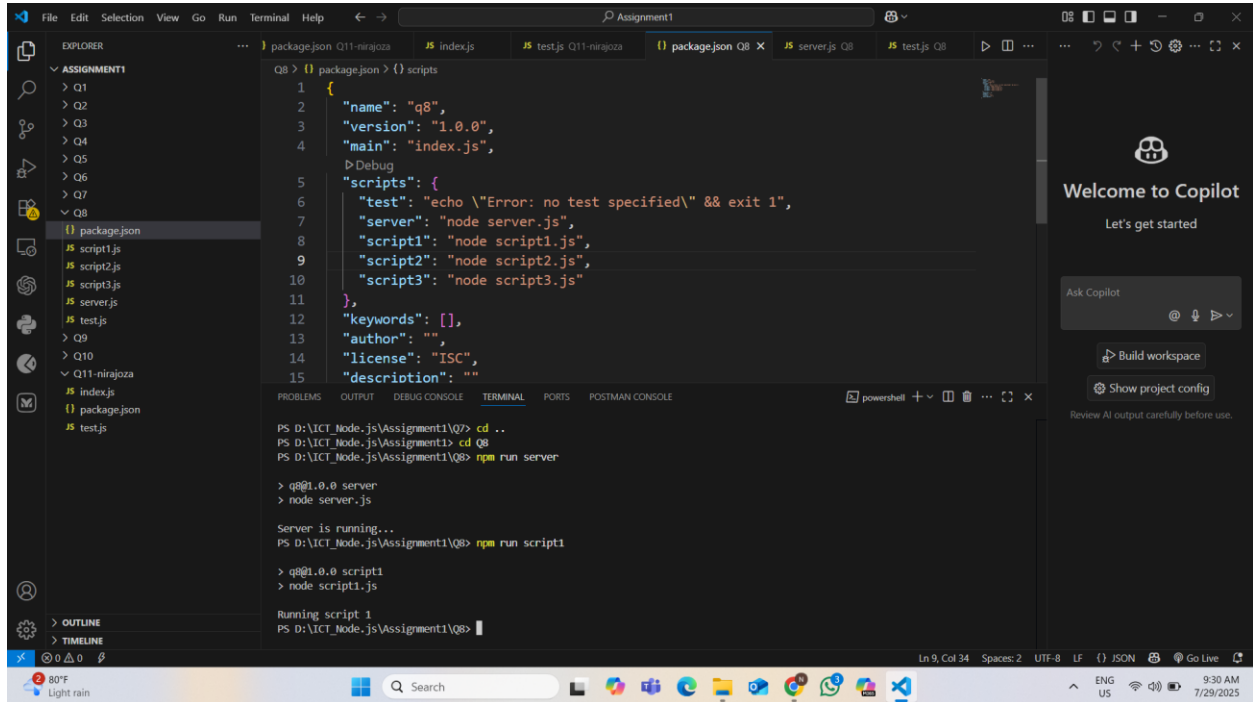
Below the code editor is the TERMINAL pane, which shows the command `node fetch-google.js` being executed. The output displays the first 500 characters of the Google homepage HTML:

```
PS D:\ICT_Node.js\Assignment1\Q7> node fetch-google.js
Fetched Google Home Page (partial content):
Fetched Google Home Page (partial content):
<doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-IN"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/google/1x/google_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="CjxIDACPJFoabxKE2r662Q">(function(){var _g={kEI:'CkeIalSyCe3V1sQPt6Waaqo',kEXPI:'0,18167,184687,2,35,3497356,702,435,302126,236535,14112,34679,390923,219814,18645,45171,6414,11186,5230576,36812642,25381059,22917,4
```

On the right side of the editor, the Copilot sidebar is visible with the text "Welcome to Copilot" and "Let's get started".

8. Set a server script, a test script and 3 user defined scripts in package.json file in your nodejs application.

ScreenShot



The screenshot shows a Visual Studio Code editor window with a project named "Assignment1". The Explorer sidebar on the left shows a file structure with a "Q8" folder containing "package.json", "index.js", "script1.js", "script2.js", "script3.js", "server.js", "test.js", and "test.js". The main editor area displays the "package.json" file for the "Q8" folder. The file contains the following JSON:

```
{
  "name": "q8",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1",
    "server": "node server.js",
    "script1": "node script1.js",
    "script2": "node script2.js",
    "script3": "node script3.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

The bottom panel shows the "TERMINAL" tab with a PowerShell prompt. The commands and output are as follows:

```
PS D:\ICT_Node.js\Assignment1\Q8> cd ..
PS D:\ICT_Node.js\Assignment1> cd Q8
PS D:\ICT_Node.js\Assignment1\Q8> npm run server

> q8@1.0.0 server
> node server.js

Server is running...
PS D:\ICT_Node.js\Assignment1\Q8> npm run script1

> q8@1.0.0 script1
> node script1.js

Running script 1
PS D:\ICT_Node.js\Assignment1\Q8>
```

The right sidebar shows the "Welcome to Copilot" panel with options to "Let's get started", "Ask Copilot", "Build workspace", and "Show project config". The status bar at the bottom indicates the file is "Ln 9, Col 34", "Spaces: 2", "UTF-8", "LF", "JSON", and "Go Live" is active. The system tray shows a temperature of 80°F, light rain, and the time 9:30 AM on 7/29/2025.

9. A program which calls useful functions in fs module.

ScreenShot

The screenshot shows the Visual Studio Code editor with a project named "Assignment1". The Explorer sidebar on the left shows the file structure, including a folder "ASSIGNMENT1" with subfolders Q1 through Q9, and files "example.txt", "fs-demo.js", "package.json", "index.js", "package.json", and "test.js". The main editor displays the code for "fs-demo.js", which is a JavaScript program using the fs module to create a folder, write a file, and append content. The code is as follows:

```
1 const fs = require("fs");
2 const path = require("path");
3
4 const dirPath = path.join(__dirname, "myfolder");
5 if (!fs.existsSync(dirPath)) {
6   fs.mkdirSync(dirPath);
7   console.log("Folder created: myfolder");
8 }
9
10 const filePath = path.join(dirPath, "example.txt");
11 fs.writeFile(filePath, "Hello from Node.js!\n", (err) => {
12   if (err) throw err;
13   console.log("File created and written to.");
14
15   fs.appendFile(filePath, "This line is appended.\n", (err) => {
16     if (err) throw err;
17   });
18 });
```

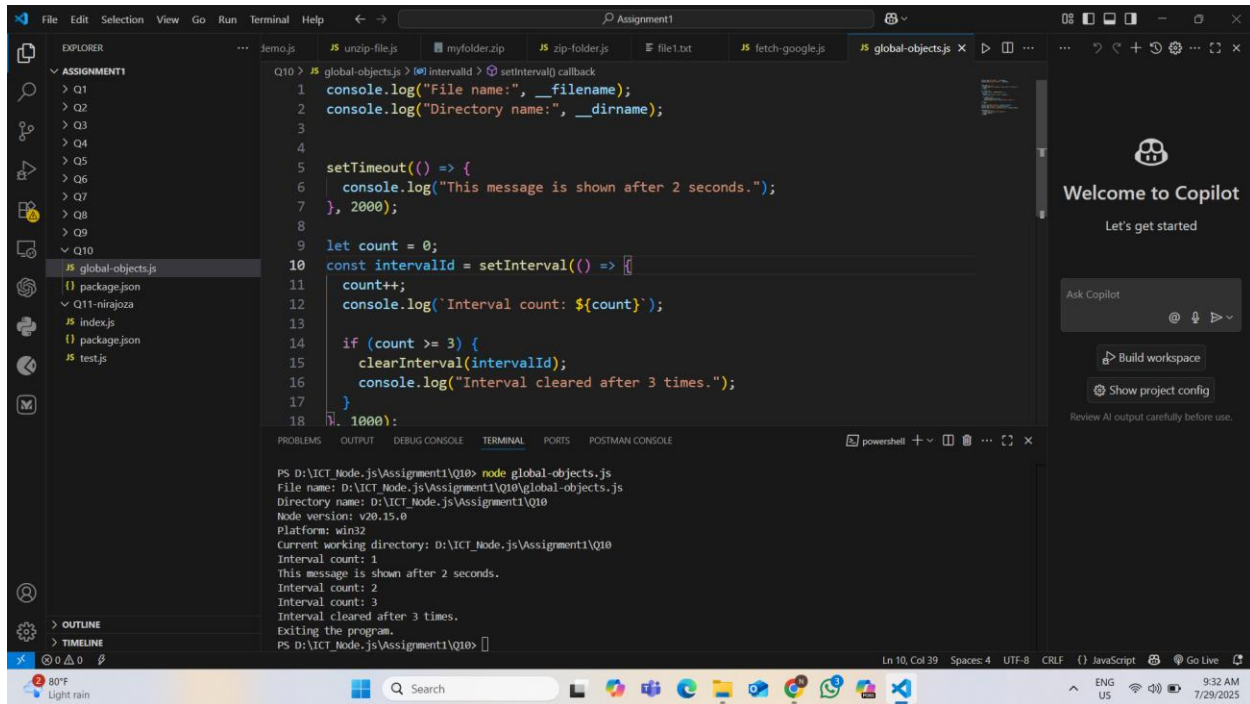
The TERMINAL pane at the bottom shows the output of running the script:

```
Running script 1
PS D:\ICT_Node.js\Assignment1\Q9> cd ..
PS D:\ICT_Node.js\Assignment1> cd Q9
PS D:\ICT_Node.js\Assignment1\Q9> node fs-demo.js
Folder created: myfolder
File created and written to.
Content appended.
File renamed to new-example.txt
Files in folder: [ 'new-example.txt' ]
File content:
Hello from Node.js!
This line is appended.
File deleted: new-example.txt
Folder removed: myfolder
PS D:\ICT_Node.js\Assignment1\Q9>
```

The right sidebar shows the "Welcome to Copilot" message, "Let's get started", and buttons for "Ask Copilot", "Build workspace", and "Show project config". The status bar at the bottom shows the file is "Ln 9, Col 1", "Spaces: 4", "UTF-8", "CRLF", and "JavaScript". The system tray at the bottom shows the temperature as "80°F", "Light rain", and the time as "9:30 AM 7/29/2025".

10. A program which uses global objects in nodejs.

ScreenShot



The screenshot shows a Visual Studio Code editor with a project named "Assignment1". The Explorer sidebar on the left shows a file structure with a folder "ASSIGNMENT1" containing files "Q1", "Q2", "Q3", "Q4", "Q5", "Q6", "Q7", "Q8", "Q9", and "Q10". The "global-objects.js" file is selected. The code in the editor is as follows:

```
Q10 > JS global-objects.js > (0) intervalId > (0) setInterval() callback
1 console.log("File name:", __filename);
2 console.log("Directory name:", __dirname);
3
4
5 setTimeout(() => {
6   console.log("This message is shown after 2 seconds.");
7 }, 2000);
8
9 let count = 0;
10 const intervalId = setInterval(() => {
11   count++;
12   console.log(`Interval count: ${count}`);
13
14   if (count >= 3) {
15     clearInterval(intervalId);
16     console.log("Interval cleared after 3 times.");
17   }
18 }, 1000);
```

The terminal at the bottom shows the output of running the program:

```
PS D:\ICT_Node.js\Assignment1\Q10> node global-objects.js
File name: D:\ICT_Node.js\Assignment1\Q10\global-objects.js
Directory name: D:\ICT_Node.js\Assignment1\Q10
Node version: v20.15.0
Platform: win32
Current working directory: D:\ICT_Node.js\Assignment1\Q10
Interval count: 1
This message is shown after 2 seconds.
Interval count: 2
Interval count: 3
Interval cleared after 3 times.
Exiting the program.
PS D:\ICT_Node.js\Assignment1\Q10>
```

The right sidebar shows the "Welcome to Copilot" panel with options like "Let's get started", "Ask Copilot", "Build workspace", and "Show project config". The status bar at the bottom indicates the current line and column (Ln 10, Col 39), encoding (UTF-8), and language (JavaScript).

11. Develop a useful package and publish it on npmjs.com

ScreenShot

