

Roll NO:30

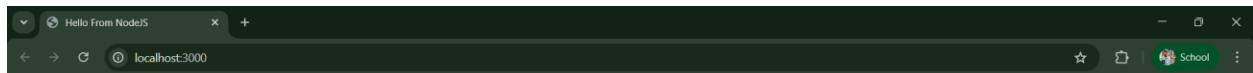
Name:Mistri Vatsal Rakeshbhai

Div:ICT

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



AJAX call to /gethello

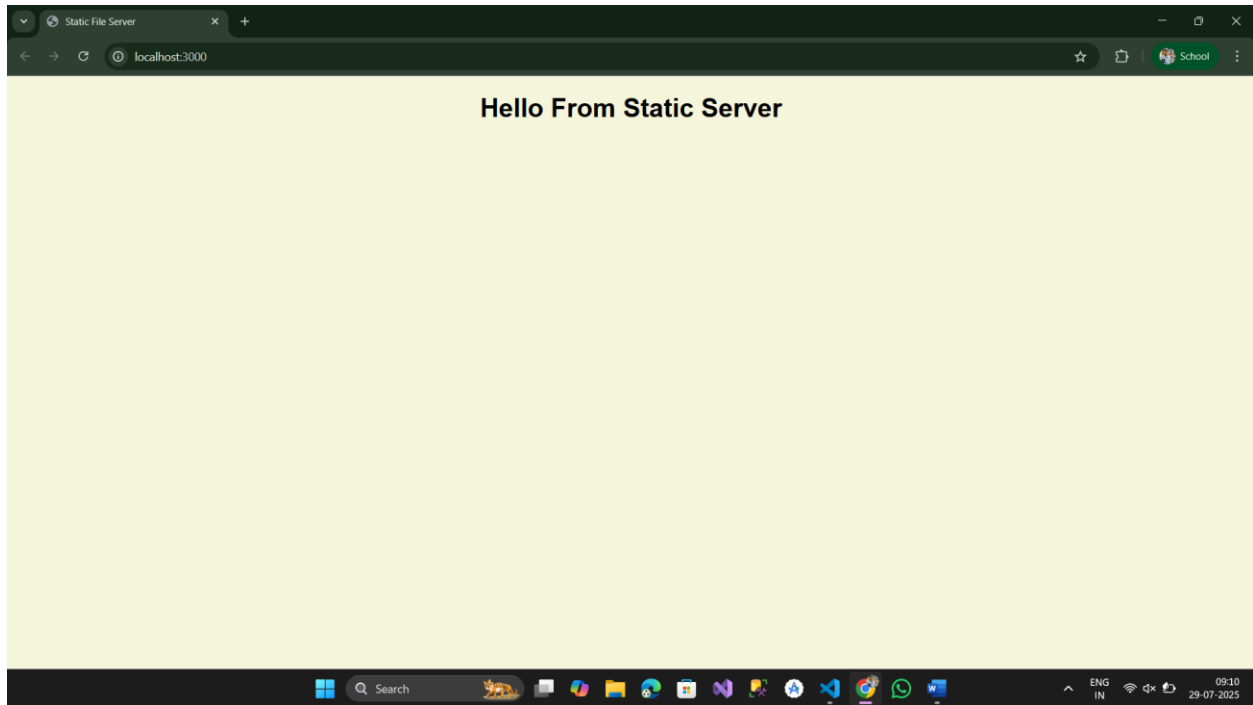
[Click Here](#)

Hello Node!!



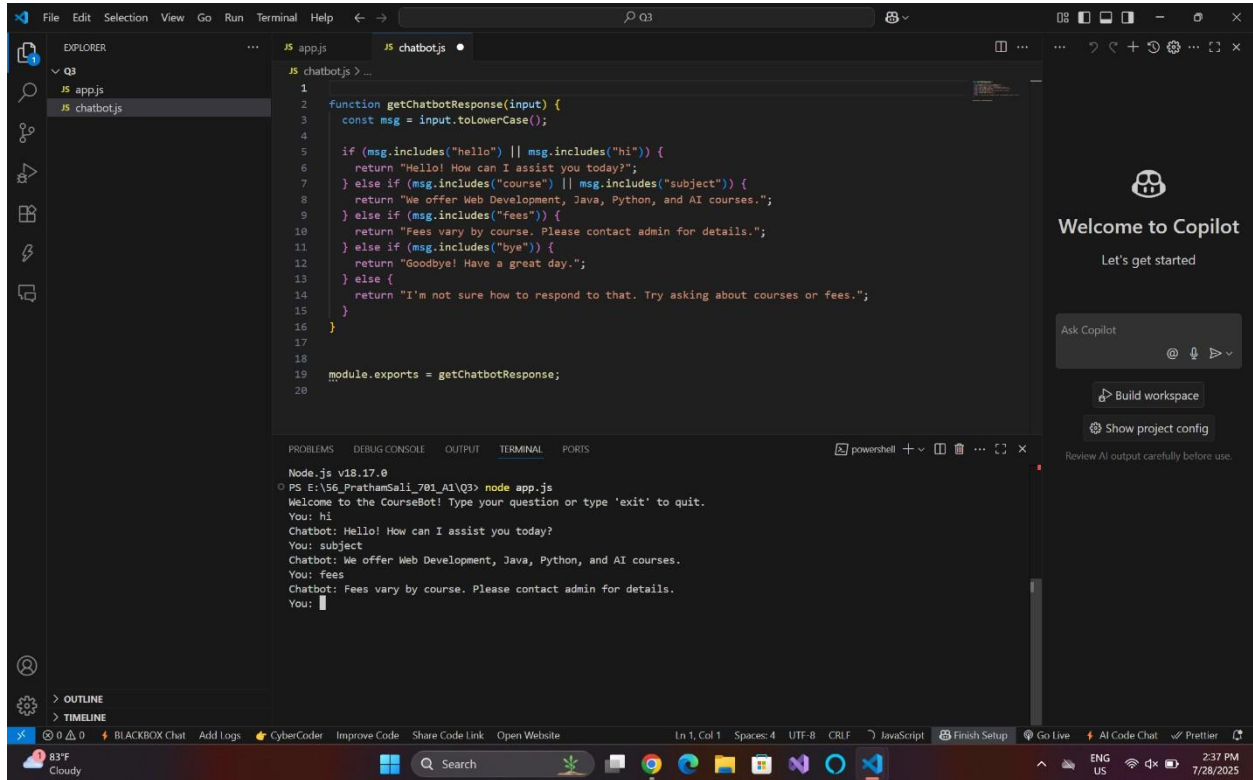
2) Develop a web server which serves static resources.

Screenshot



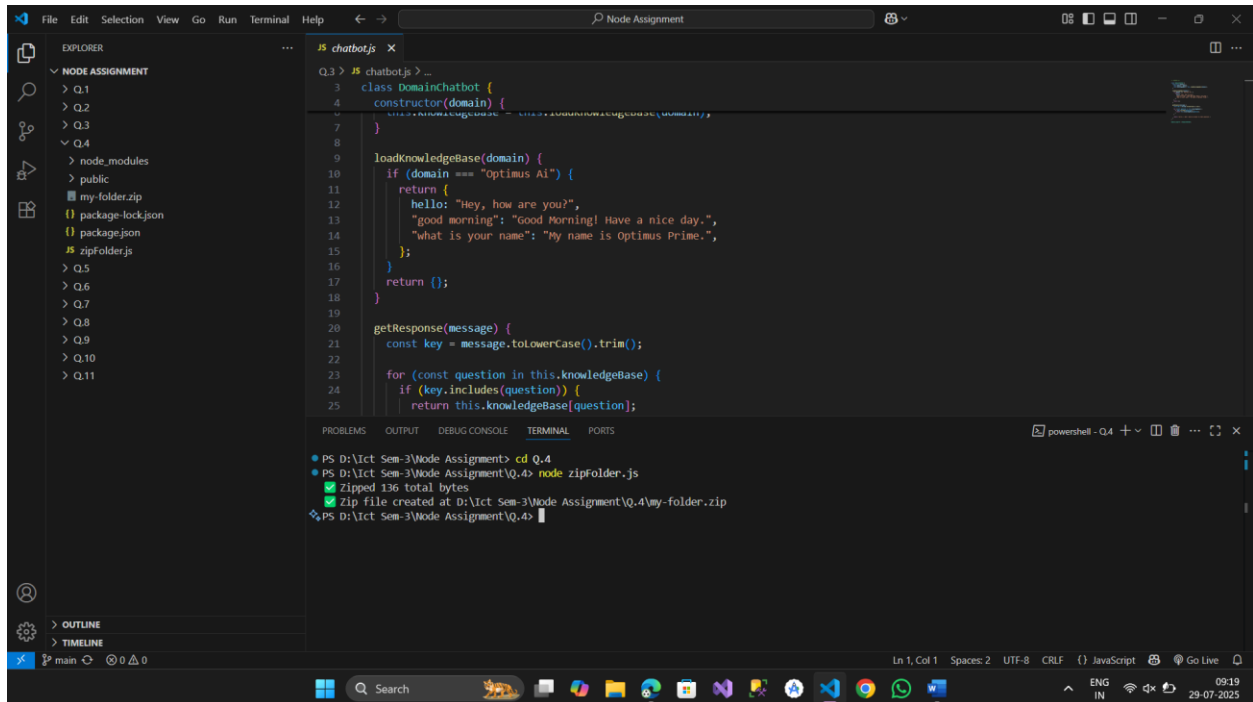
3) Develop a module for domain specific chatbot and use it in a command line application.

ScreenShot



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

Screenshot



The screenshot shows a Visual Studio Code editor window with a file named `chatbot.js` open. The file contains a JavaScript class `DomainChatbot` with a constructor and two methods: `loadKnowledgeBase` and `getResponse`. The `loadKnowledgeBase` method checks if the domain is "Optimus AI" and returns a greeting and a name. The `getResponse` method takes a message and returns a response based on the knowledge base.

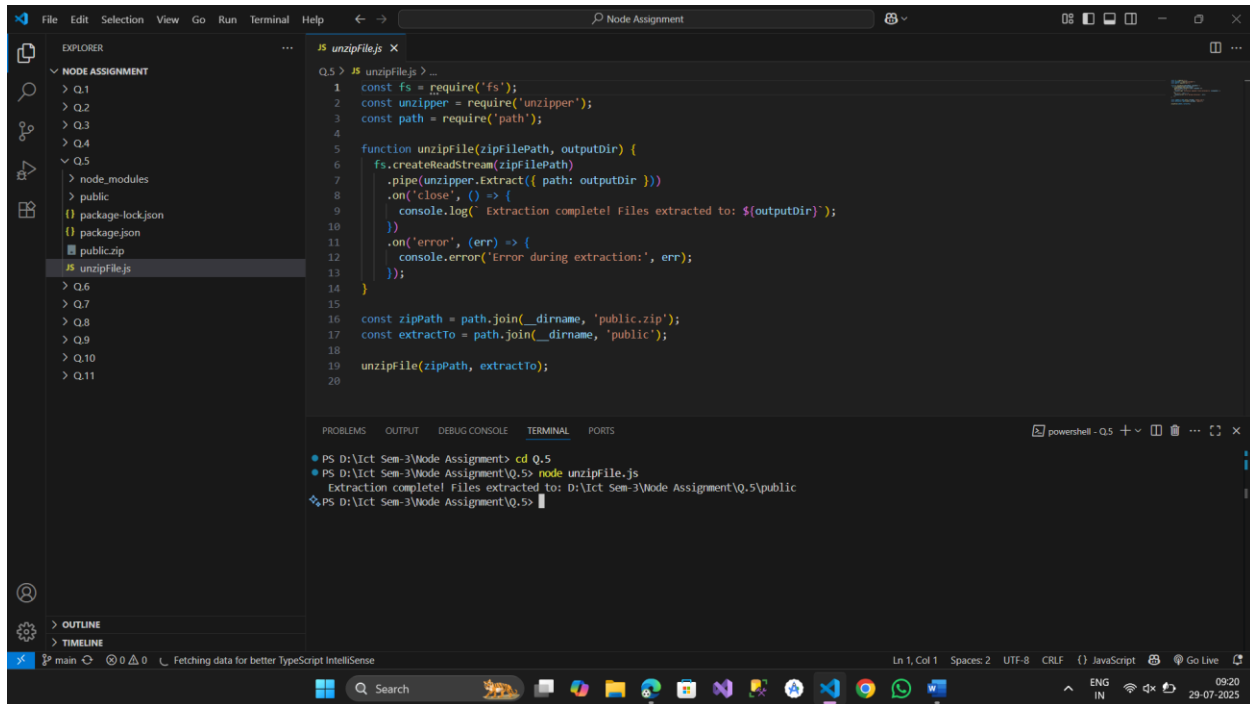
```
Q.3 > JS chatbot.js > ...
3  class DomainChatbot {
4    constructor(domain) {
5      this.knowledgeBase = {
6        'hello': 'Hey, how are you?',
7        'good morning': 'Good Morning! Have a nice day.',
8        'what is your name': 'My name is Optimus Prime.'
9      };
10   }
11   loadKnowledgeBase(domain) {
12     if (domain === "Optimus AI") {
13       return {
14         hello: "Hey, how are you?",
15         "good morning": "Good Morning! Have a nice day.",
16         "what is your name": "My name is Optimus Prime."
17       };
18     }
19     return {};
20   }
21   getResponse(message) {
22     const key = message.toLowerCase().trim();
23     for (const question in this.knowledgeBase) {
24       if (key.includes(question)) {
25         return this.knowledgeBase[question];
26       }
27     }
28     return "I don't know the answer to that question.";
29   }
30 }
```

The terminal window at the bottom shows the following commands and output:

```
PS D:\Ict Sem-3\Node Assignment> cd Q.4
PS D:\Ict Sem-3\Node Assignment\Q.4> node zipfolder.js
Zipped 136 total bytes
Zip file created at D:\Ict Sem-3\Node Assignment\Q.4\my-folder.zip
PS D:\Ict Sem-3\Node Assignment\Q.4>
```

5) Write a program to extract a zip file.

Screenshot



The screenshot displays the Visual Studio Code editor with a project named 'Node Assignment'. The Explorer sidebar on the left shows a file structure with folders 'Q.1' through 'Q.11', and files 'package-lock.json', 'package.json', and 'public.zip'. The file 'unzipFile.js' is selected and open in the editor. The code in 'unzipFile.js' is as follows:

```
1 const fs = require('fs');
2 const unzipper = require('unzipper');
3 const path = require('path');
4
5 function unzipFile(zipfilePath, outputDir) {
6   fs.createReadStream(zipfilePath)
7     .pipe(unzipper.Extract({ path: outputDir }))
8     .on('close', () => {
9       console.log('Extraction complete! Files extracted to: ${outputDir}');
10     })
11     .on('error', (err) => {
12       console.error('Error during extraction:', err);
13     });
14 }
15
16 const zipPath = path.join(__dirname, 'public.zip');
17 const extractTo = path.join(__dirname, 'public');
18
19 unzipFile(zipPath, extractTo);
20
```

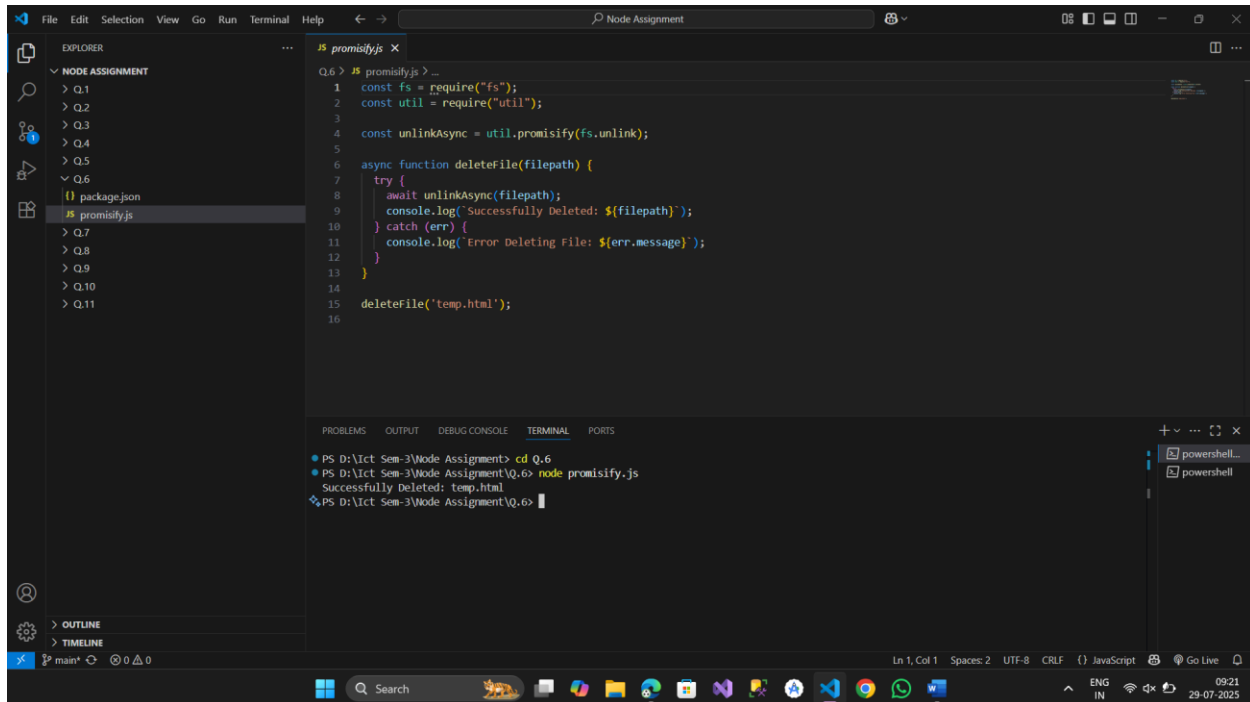
The bottom panel shows the 'TERMINAL' tab with the following commands and output:

```
PS D:\Ict Sem-3\Node Assignment> cd Q.5
PS D:\Ict Sem-3\Node Assignment\Q.5> node unzipFile.js
Extraction complete! Files extracted to: D:\Ict Sem-3\Node Assignment\Q.5\public
PS D:\Ict Sem-3\Node Assignment\Q.5>
```

The status bar at the bottom indicates the file is 'Ln 1, Col 1', 'Spaces: 2', 'UTF-8', 'CRLF', and 'JavaScript'. The system tray shows the time as 09:20 on 29-07-2025.

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

Screenshot



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

```
1 const fs = require("fs");
2 const util = require("util");
3
4 const unlinkAsync = util.promisify(fs.unlink);
5
6 async function deleteFile(filepath) {
7   try {
8     await unlinkAsync(filepath);
9     console.log("Successfully Deleted: ${filepath}");
10  } catch (err) {
11    console.log("Error Deleting File: ${err.message}");
12  }
13 }
14
15 deleteFile('temp.html');
16
```

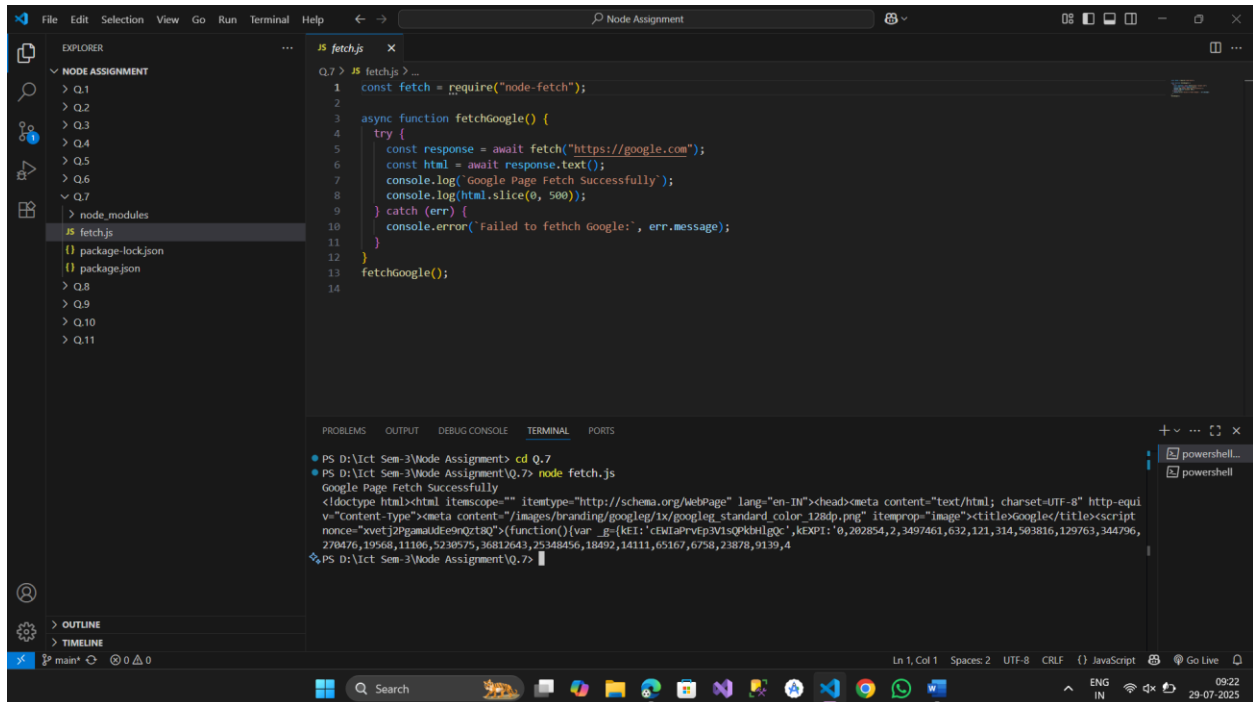
The Explorer sidebar on the left shows a project structure with a folder named `Q.6` containing files `package.json` and `promisify.js`. The terminal at the bottom shows the following commands and output:

```
PS D:\Ict Sem-3\Node Assignment> cd Q.6
PS D:\Ict Sem-3\Node Assignment\Q.6> node promisify.js
Successfully Deleted: temp.html
PS D:\Ict Sem-3\Node Assignment\Q.6>
```

The status bar at the bottom indicates the current file is `promisify.js`, line 1, column 1, using UTF-8 encoding and CRLF line endings.

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 file named `fetch.js` open. The file contains the following JavaScript code:

```
Q.7 > JS fetch.js > ...
1  const fetch = require("node-fetch");
2
3  async function fetchGoogle() {
4    try {
5      const response = await fetch("https://google.com");
6      const html = await response.text();
7      console.log("Google Page Fetch Successfully");
8      console.log(html.slice(0, 500));
9    } catch (err) {
10     console.error("Failed to fetch Google:", err.message);
11   }
12 }
13 fetchGoogle();
14
```

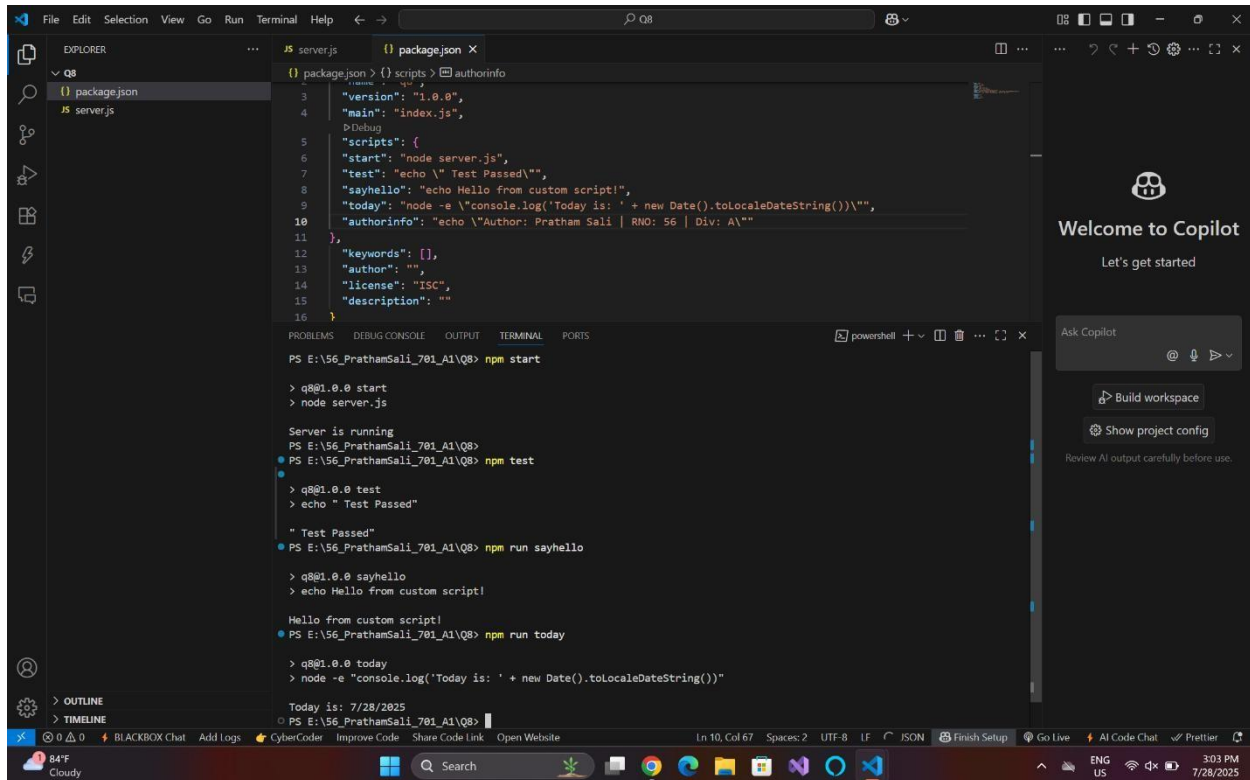
The Explorer sidebar on the left shows a project structure with a folder named `node_modules` and a file named `fetch.js`. The Terminal panel at the bottom shows the command `node fetch.js` being executed, resulting in the following output:

```
PS D:\Ict Sem-3\Node Assignment> cd Q.7
PS D:\Ict Sem-3\Node Assignment\Q.7> node fetch.js
Google Page Fetch Successfully
<!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/googlep/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="xvetj2PgamaUde9nQzt8Q">(function(){var _g={key:'cEwIaPrvEp3V1sQpKbHlgQc',keyXPI:'0,202854,2,3497461,632,121,314,503816,129763,344796,270476,19568,11106,5230575,36812643,75348456,18492,14111,65167,6758,23878,9139,4
```

The status bar at the bottom indicates the current file is `main.js`, the editor is in `UTF-8` encoding, and the language is `JavaScript`. The system clock shows the time as 09:22 on 29-07-2025.

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 with a project named 'Q8'. The Explorer sidebar on the left shows a file named 'package.json'. The main editor area displays the contents of 'package.json', which includes a 'scripts' section with the following configuration:

```
{
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "start": "node server.js",
    "test": "echo \" Test Passed\"",
    "sayhello": "echo Hello from custom script!",
    "today": "node -e \"console.log('Today is: ' + new Date().toLocaleDateString())\"",
    "authorinfo": "echo \"Author: Pratham Sali | RNO: 56 | Div: A\""
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

Below the editor, the TERMINAL panel shows the execution of several npm commands in a PowerShell session:

```
PS E:\56_PrathamSali_701_A1\Q8> npm start
> q8@1.0.0 start
> node server.js

Server is running
PS E:\56_PrathamSali_701_A1\Q8>
PS E:\56_PrathamSali_701_A1\Q8> npm test
> q8@1.0.0 test
> echo " Test Passed"

" Test Passed"
PS E:\56_PrathamSali_701_A1\Q8> npm run sayhello
> q8@1.0.0 sayhello
> echo Hello from custom script!

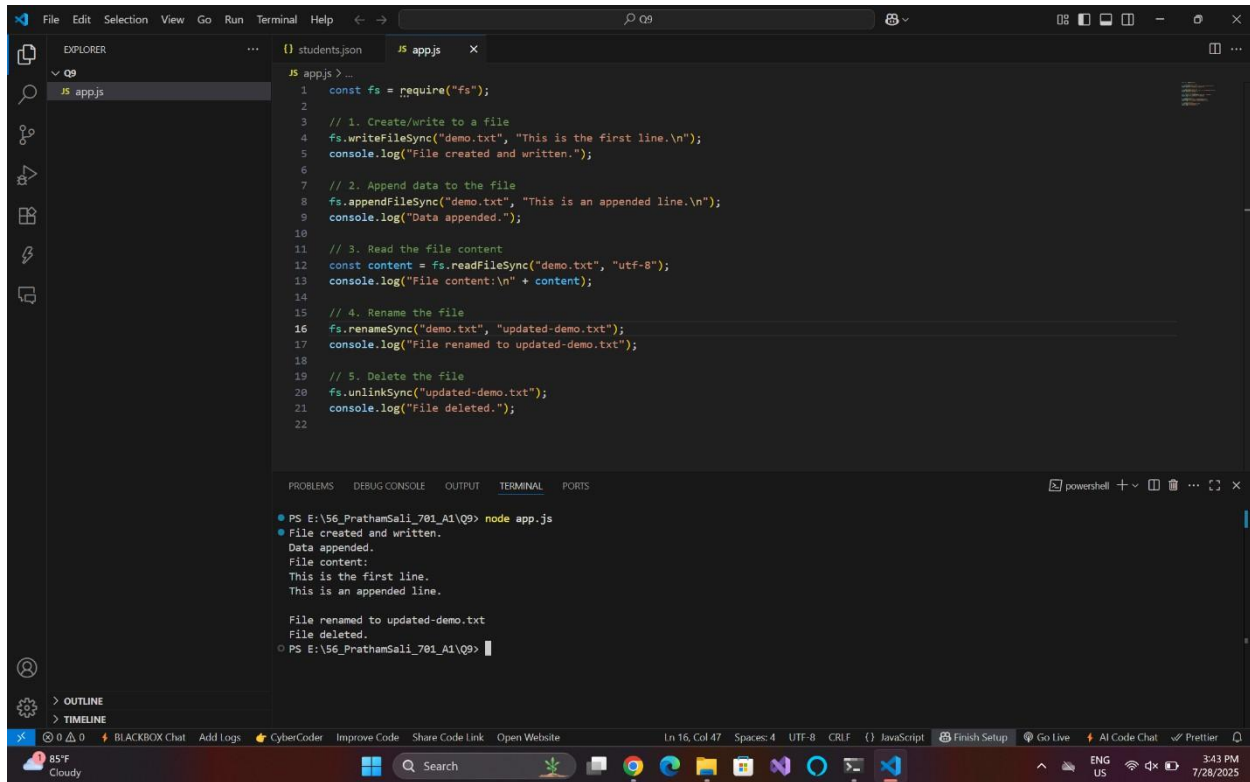
Hello from custom script!
PS E:\56_PrathamSali_701_A1\Q8> npm run today
> q8@1.0.0 today
> node -e "console.log('Today is: ' + new Date().toLocaleDateString())"

Today is: 7/28/2025
PS E:\56_PrathamSali_701_A1\Q8>
```

On the right side of the interface, the 'Welcome to Copilot' sidebar is visible, offering options like 'Let's get started', 'Ask Copilot', 'Build workspace', and 'Show project config'.

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

Screenshot



The screenshot shows a Visual Studio Code editor with a JavaScript file named `app.js` open. The code uses the `fs` module to perform various file operations. The terminal output shows the execution of `node app.js` and the resulting log messages.

```
1 const fs = require("fs");
2
3 // 1. Create/write to a file
4 fs.writeFileSync("demo.txt", "This is the first line.\n");
5 console.log("File created and written.");
6
7 // 2. Append data to the file
8 fs.appendFileSync("demo.txt", "This is an appended line.\n");
9 console.log("Data appended.");
10
11 // 3. Read the file content
12 const content = fs.readFileSync("demo.txt", "utf-8");
13 console.log("File content:\n" + content);
14
15 // 4. Rename the file
16 fs.renameSync("demo.txt", "updated-demo.txt");
17 console.log("File renamed to updated-demo.txt");
18
19 // 5. Delete the file
20 fs.unlinkSync("updated-demo.txt");
21 console.log("File deleted.");
22
```

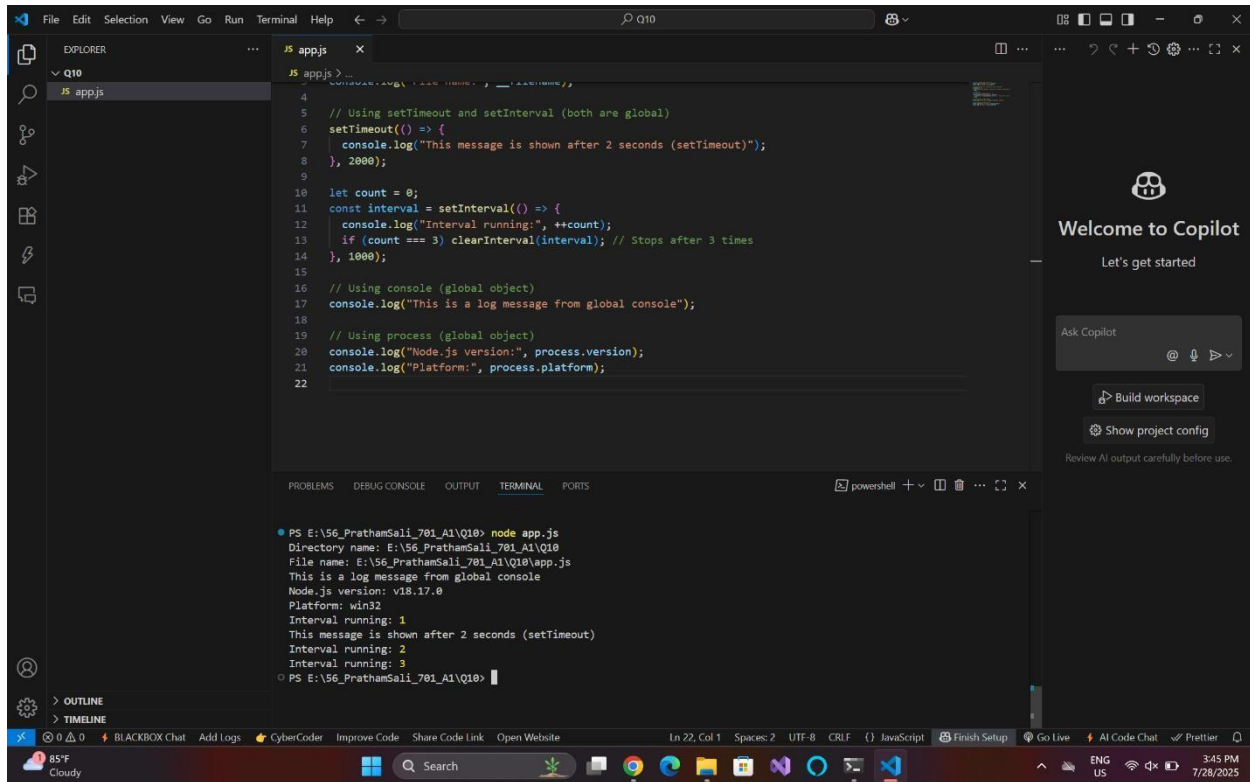
Terminal Output:

```
PS E:\S6_PrathamSali_701_A1\Q9> node app.js
File created and written.
Data appended.
File content:
This is the first line.
This is an appended line.

File renamed to updated-demo.txt
File deleted.
PS E:\S6_PrathamSali_701_A1\Q9>
```

10. A program which uses global objects in nodejs.

Screenshot



The screenshot displays the Visual Studio Code editor with a file named `app.js` open. The code uses `setTimeout` and `setInterval` (both global objects) to log messages. It also uses `console` (global object) and `process` (global object) to log system information. The terminal shows the output of running `node app.js`, including the directory name, file name, Node.js version (v18.17.0), platform (win32), and the execution of the `setTimeout` and `setInterval` functions.

```
4 // Using setTimeout and setInterval (both are global)
5 // Using setTimeout and setInterval (both are global)
6 setTimeout(() => {
7   console.log("This message is shown after 2 seconds (setTimeout)");
8 }, 2000);
9
10 let count = 0;
11 const interval = setInterval(() => {
12   console.log("Interval running:", ++count);
13   if (count === 3) clearInterval(interval); // Stops after 3 times
14 }, 1000);
15
16 // Using console (global object)
17 console.log("This is a log message from global console");
18
19 // Using process (global object)
20 console.log("Node.js version:", process.version);
21 console.log("Platform:", process.platform);
22
```

Terminal Output:

```
PS E:\S6_PrathamSali_701_A1\Q10> node app.js
Directory name: E:\S6_PrathamSali_701_A1\Q10
File name: E:\S6_PrathamSali_701_A1\Q10\app.js
This is a log message from global console
Node.js version: v18.17.0
Platform: win32
Interval running: 1
This message is shown after 2 seconds (setTimeout)
Interval running: 2
Interval running: 3
PS E:\S6_PrathamSali_701_A1\Q10>
```

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

Screenshot

