

Assignment 2

1. Develop a user registration form (fields: username, password, confirm password, email, gender, hobbies, upload profile pic, upload other pics) (use text box, radio, checkboxes, file upload) using Express. Form should also contain file upload (single, multiple) with validations.

- In case of invalid data, display errors along with all previous field values.
- In case of all valid fields, display all data and images in a well formatted tabular form. and allow the user to download that file using Express. Develop route for file download. (Use express, ejs, express-validator)

Output :-

Registration Form

Username:

Email:

Password:

Confirm Password:

Gender: ☒ Male ☐ Female

Hobbies: ☐ Reading ☐ Sports ☒ Music

Profile Pic: 3b035002-0...991966c.jpg

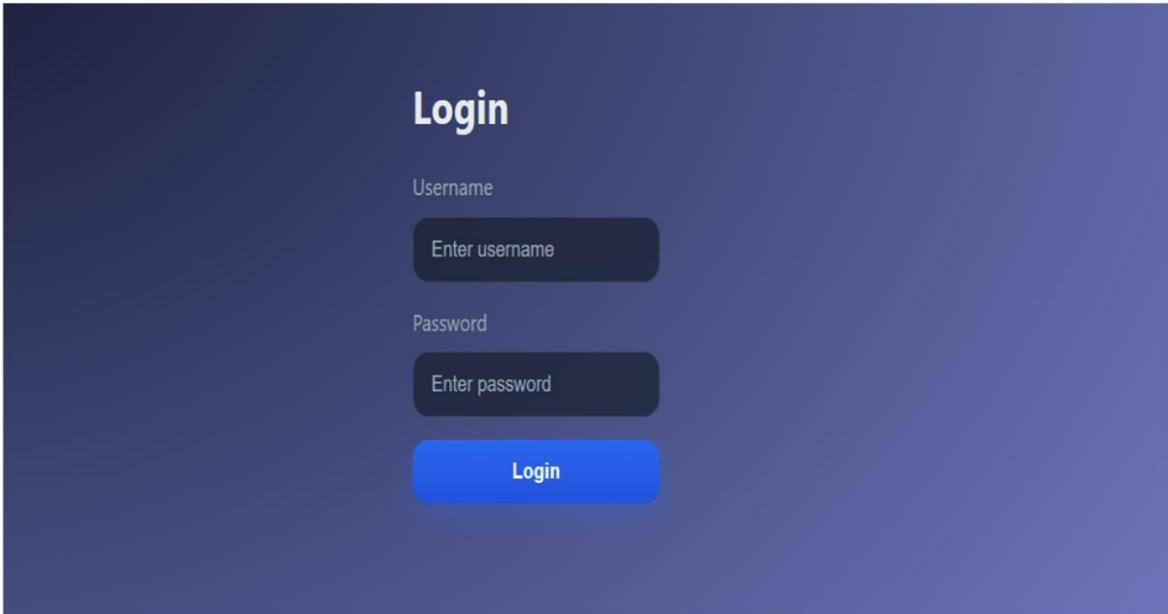
Other Pics: 2.jpg

Username	Nayan
Email	nayanprakashbhai@gmail.com
Gender	Male
Hobbies	Music
Profile Pic	
Other Pics	

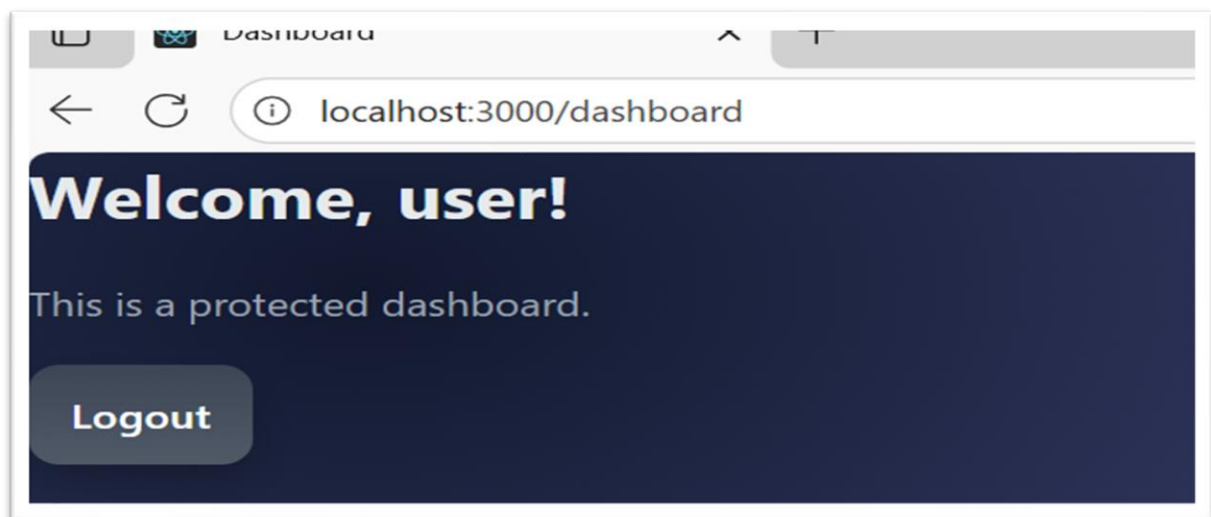
[Download User Data](#)

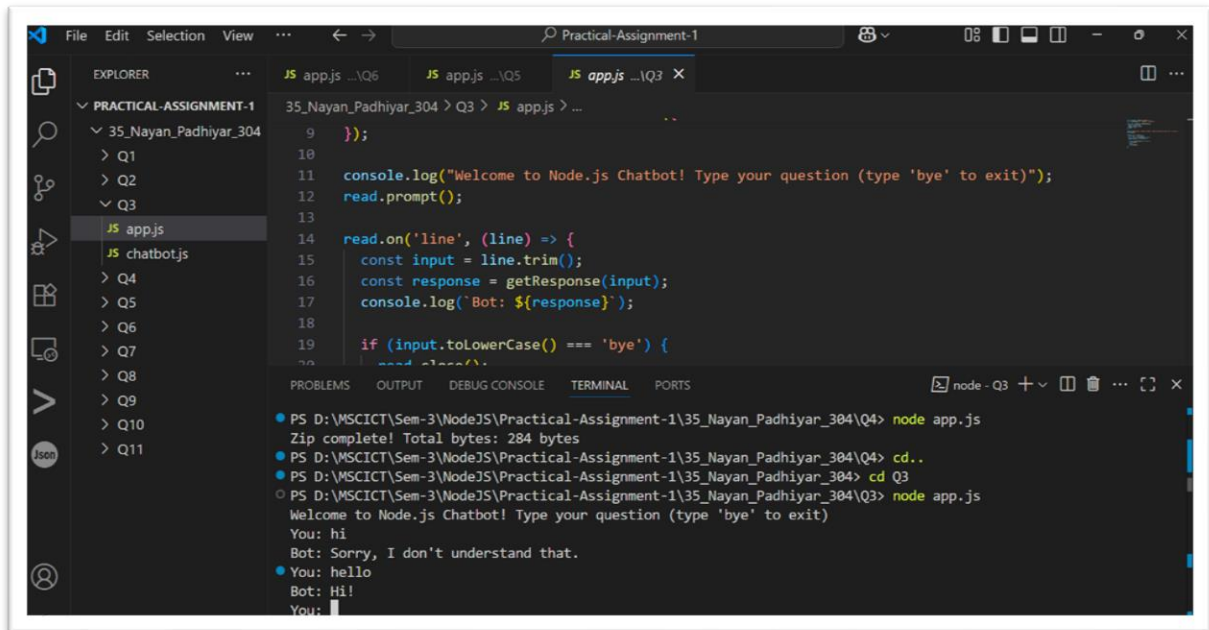
2. Express Login application with file session store.

Output :-



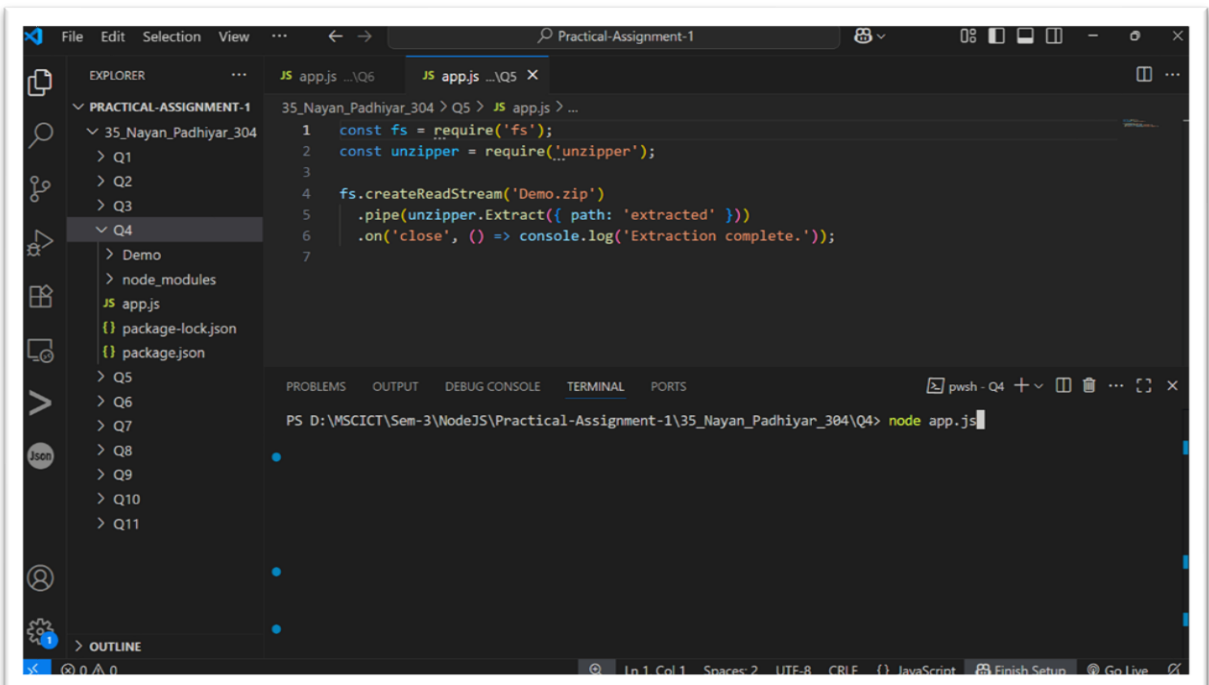
A screenshot of a web application's login page. The page has a dark blue background. At the top center, the word "Login" is displayed in a large, white, sans-serif font. Below it, the label "Username" is shown in a smaller, light gray font. Underneath the label is a dark gray rounded rectangular input field containing the placeholder text "Enter username". Below the input field is the label "Password" in a smaller, light gray font. Underneath the label is another dark gray rounded rectangular input field containing the placeholder text "Enter password". At the bottom center of the form is a bright blue rounded rectangular button with the word "Login" in white text.



33) Express Login application with redis session store.**Output :-**

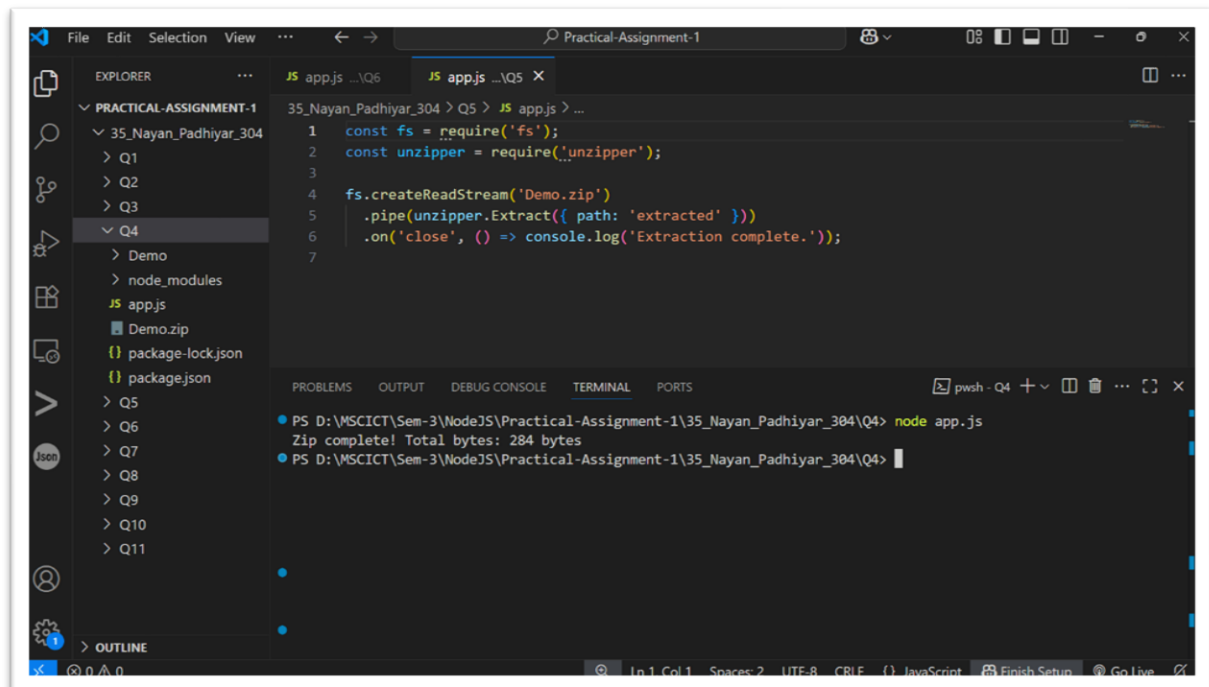
```
File Edit Selection View ... Practical-Assignment-1
EXPLORER
PRACTICAL-ASSIGNMENT-1
  35_Nayan_Padhiyar_304
    > Q1
    > Q2
    > Q3
    JS app.js
    JS chatbot.js
    > Q4
    > Q5
    > Q6
    > Q7
    > Q8
    > Q9
    > Q10
    > Q11

35_Nayan_Padhiyar_304 > Q3 > JS app.js > ...
9   });
10
11  console.log("Welcome to Node.js Chatbot! Type your question (type 'bye' to exit)");
12  read.prompt();
13
14  read.on('line', (line) => {
15    const input = line.trim();
16    const response = getResponse(input);
17    console.log(`Bot: ${response}`);
18
19    if (input.toLowerCase() === 'bye') {
20      read.close();
21    }
22  });
23
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
node - Q3 + - - - - -
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4> node app.js
Zip complete! Total bytes: 284 bytes
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4> cd..
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304> cd Q3
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q3> node app.js
Welcome to Node.js Chatbot! Type your question (type 'bye' to exit)
You: hi
Bot: Sorry, I don't understand that.
You: hello
Bot: Hi!
You:
```

4. Write a program to create a compressed zip file for a folder.**Output :-**

```
File Edit Selection View ... Practical-Assignment-1
EXPLORER
PRACTICAL-ASSIGNMENT-1
  35_Nayan_Padhiyar_304
    > Q1
    > Q2
    > Q3
    > Q4
    Demo
    > node_modules
    JS app.js
    package-lock.json
    package.json
    > Q5
    > Q6
    > Q7
    > Q8
    > Q9
    > Q10
    > Q11

35_Nayan_Padhiyar_304 > Q5 > JS app.js > ...
1  const fs = require('fs');
2  const unzipper = require('unzipper');
3
4  fs.createReadStream('Demo.zip')
5    .pipe(unzipper.Extract({ path: 'extracted' }))
6    .on('close', () => console.log('Extraction complete.));
7
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
pwsh - Q4 + - - - - -
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4> node app.js
```



The screenshot shows the Visual Studio Code editor with a project named 'Practical-Assignment-1'. The Explorer sidebar on the left shows a folder structure for '35_Nayan_Padhiyar_304' containing subfolders Q1 through Q11, and files 'app.js', 'Demo.zip', 'package-lock.json', and 'package.json'. The file 'app.js' is open in the editor, displaying the following JavaScript code:

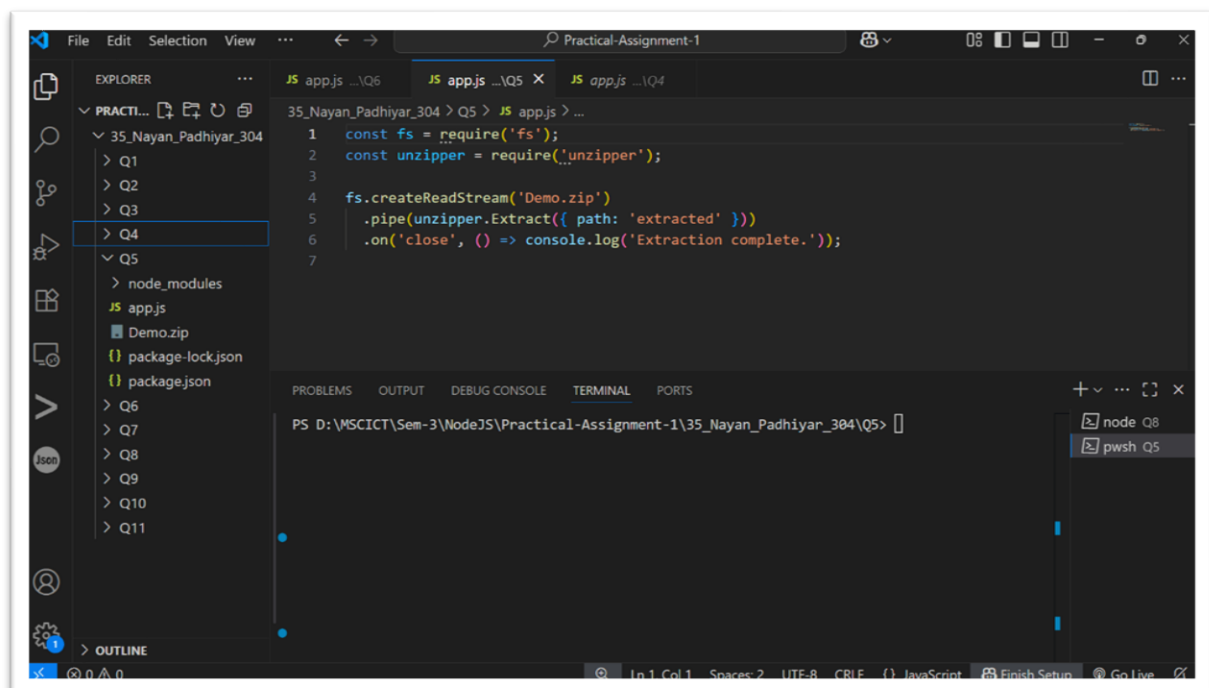
```
1 const fs = require('fs');
2 const unzipper = require('unzipper');
3
4 fs.createReadStream('Demo.zip')
5   .pipe(unzipper.Extract({ path: 'extracted' })))
6   .on('close', () => console.log('Extraction complete.'));
7
```

The TERMINAL panel at the bottom shows the command prompt output for running 'node app.js' in the directory 'D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4'. The output is:

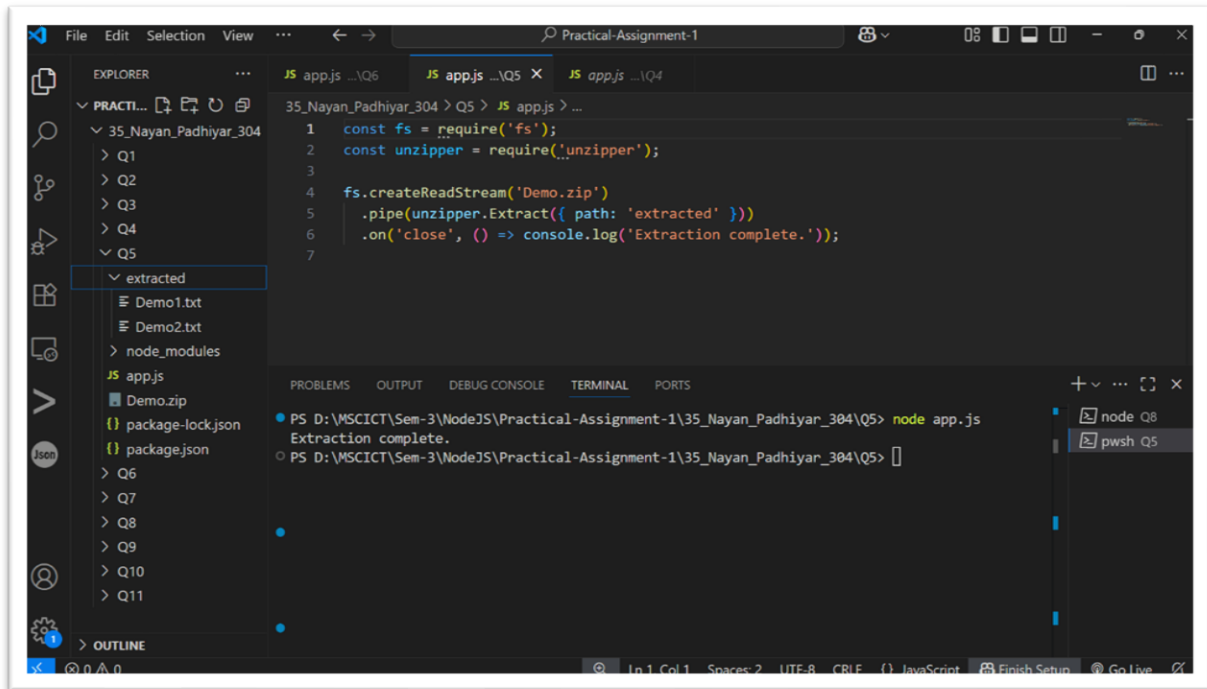
```
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4> node app.js
Zip complete! Total bytes: 284 bytes
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q4>
```

5. Write a program to extract a zip file.

Output :-



This screenshot shows the same Visual Studio Code environment. The Explorer sidebar now highlights the 'Q5' folder. The 'app.js' file is still open in the editor, showing the same code as in the first screenshot. The TERMINAL panel shows a different command prompt session, with the command 'node app.js' being entered in the directory 'D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q5'. The output is not yet visible, only the command prompt is shown.



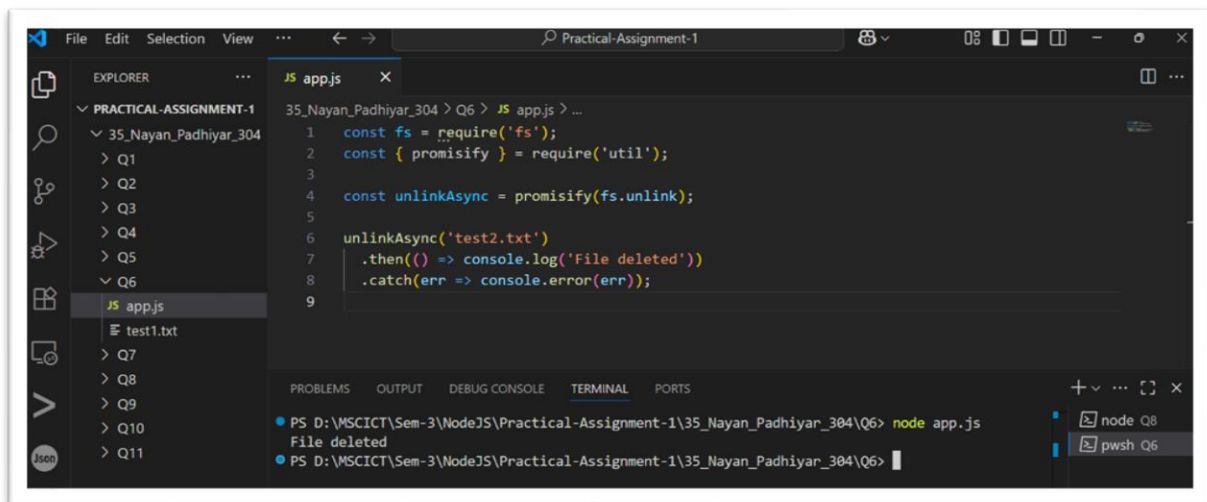
The screenshot shows a VS Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project named '35_Nayan_Padhiyar_304' with a folder 'extracted' containing 'Demo1.txt' and 'Demo2.txt'. The code editor shows a file named 'app.js' with the following code:

```
1 const fs = require('fs');
2 const unzipper = require('unzipper');
3
4 fs.createReadStream('Demo.zip')
5   .pipe(unzipper.Extract({ path: 'extracted' })))
6   .on('close', () => console.log('Extraction complete.));
7
```

The terminal at the bottom shows the command 'node app.js' being executed, resulting in the output 'Extraction complete.'.

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

Output :-



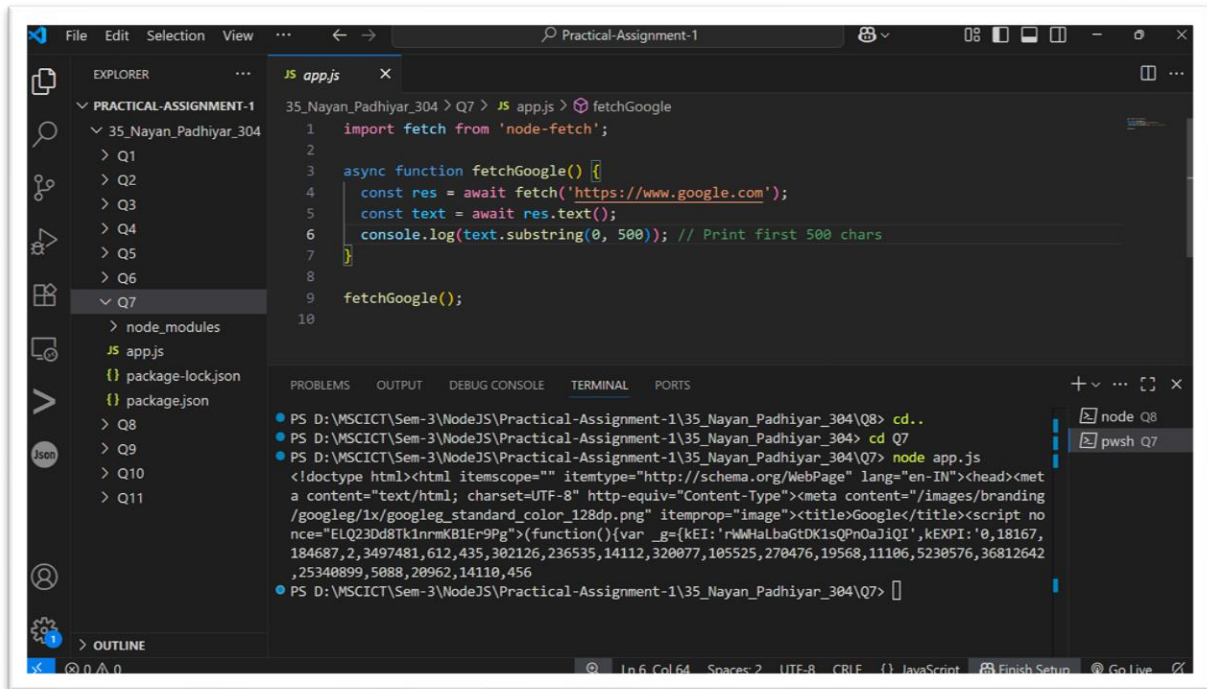
The screenshot shows a VS Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project named 'PRACTICAL-ASSIGNMENT-1' with a folder '35_Nayan_Padhiyar_304' containing a file 'test1.txt'. The code editor shows a file named 'app.js' with the following code:

```
1 const fs = require('fs');
2 const { promisify } = require('util');
3
4 const unlinkAsync = promisify(fs.unlink);
5
6 unlinkAsync('test2.txt')
7   .then(() => console.log('File deleted'))
8   .catch(err => console.error(err));
9
```

The terminal at the bottom shows the command 'node app.js' being executed, resulting in the output 'File deleted'.

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

Output :-



The screenshot shows the Visual Studio Code editor interface. The Explorer panel on the left displays the file structure of a project named 'PRACTICAL-ASSIGNMENT-1'. Inside a folder named '35_Nayan_Padhiyar_304', there is a subfolder 'Q7' containing files 'node_modules', 'app.js', 'package-lock.json', and 'package.json'. The main editor window shows the content of 'app.js', which is a JavaScript script using the 'fetch' API to retrieve data from 'https://www.google.com/'. The script defines an asynchronous function 'fetchGoogle()' that fetches the page content, extracts the first 500 characters, and logs them to the console. The terminal window at the bottom shows the execution of the script using 'node app.js' in the 'Q7' directory, displaying the HTML content of the Google homepage.

```
35_Nayan_Padhiyar_304 > Q7 > JS app.js > fetchGoogle
1  import fetch from 'node-fetch';
2
3  async function fetchGoogle() {
4      const res = await fetch('https://www.google.com');
5      const text = await res.text();
6      console.log(text.substring(0, 500)); // Print first 500 chars
7  }
8
9  fetchGoogle();
10
```

```
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q8> cd..
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304> cd Q7
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q7> node app.js
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-IN"><head><met
a content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding
/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script no
nce="ELQ23Dd8Tk1nmKB1Er9Pg">(function(){var _g={kEI:'rWwHalbaGtDK1sQPn0aJiQI',kEXPI:'0,18167,
184687,2,3497481,612,435,302126,236535,14112,320077,105525,270476,19568,11106,5230576,36812642
,25340899,5088,20962,14110,456
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q7>
```

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

Output :-

The screenshot shows a VS Code editor with a file named `package.json` open. The file contains the following JSON:

```

{
  "scripts": {
    "start": "node app.js",
    "test": "node test.js",
    "sayhello": "echo Hello from custom script!",
    "currentfile": "node -e \"console.log(process.cwd())\"",
    "userdata": "node -e \"console.log(require('os').userInfo())\""
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

```

The Explorer sidebar shows a folder named `35_Nayan_Padhiyar_304` with files `app.js`, `package.json`, and `test.js`. The `package.json` file is selected. The Terminal panel at the bottom shows the command `node -e "console.log(require('os').userInfo())"` being executed, resulting in the following output:

```

{
  uid: -1,
  gid: -1,
  username: 'NISP',
  homedir: 'C:\\Users\\DELL',
  shell: null
}

```

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

Output :-

The screenshot shows a VS Code editor with a file named `app.js` open. The file contains the following JavaScript code:

```

fs.writeFileSync('example.txt', 'Hello, this is a test file!');
fs.appendFileSync('example.txt', '\nAppended line. ');
const content = fs.readFileSync('example.txt', 'utf-8');
console.log("File Content:\n", content);
fs.renameSync('example.txt', 'rename.txt');
if (fs.existsSync('rename.txt')) {
  console.log('rename.txt exists!');
}

```

The Explorer sidebar shows a folder named `35_Nayan_Padhiyar_304` with files `app.js`, `demo.txt`, and `rename.txt`. The `app.js` file is selected. The Terminal panel at the bottom shows the command `node app.js` being executed, resulting in the following output:

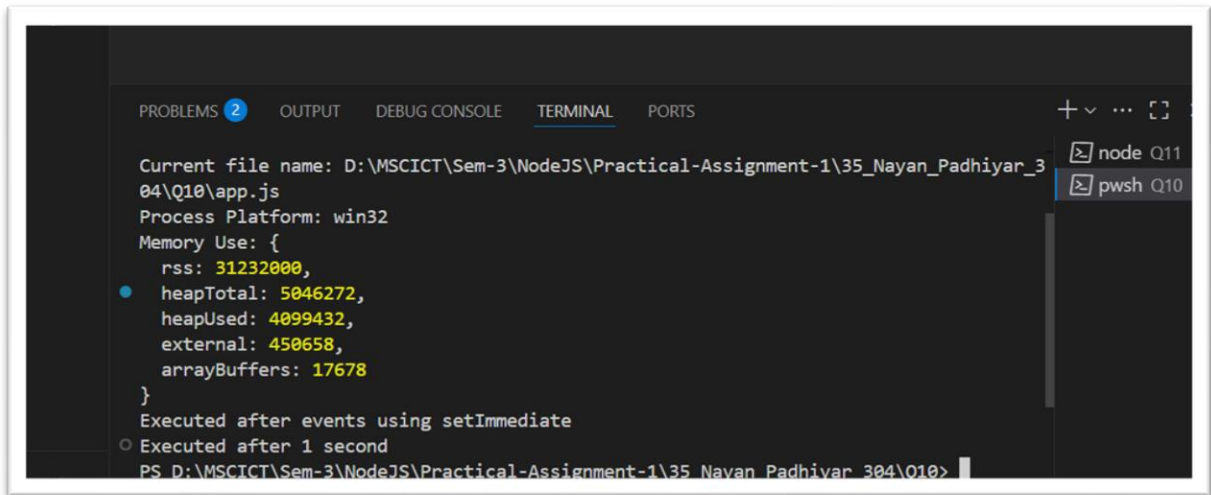
```

File Content:
Hello, this is a test file!
Appended line.
rename.txt exists!

```

10. A program which uses global objects in nodejs.

Output :-

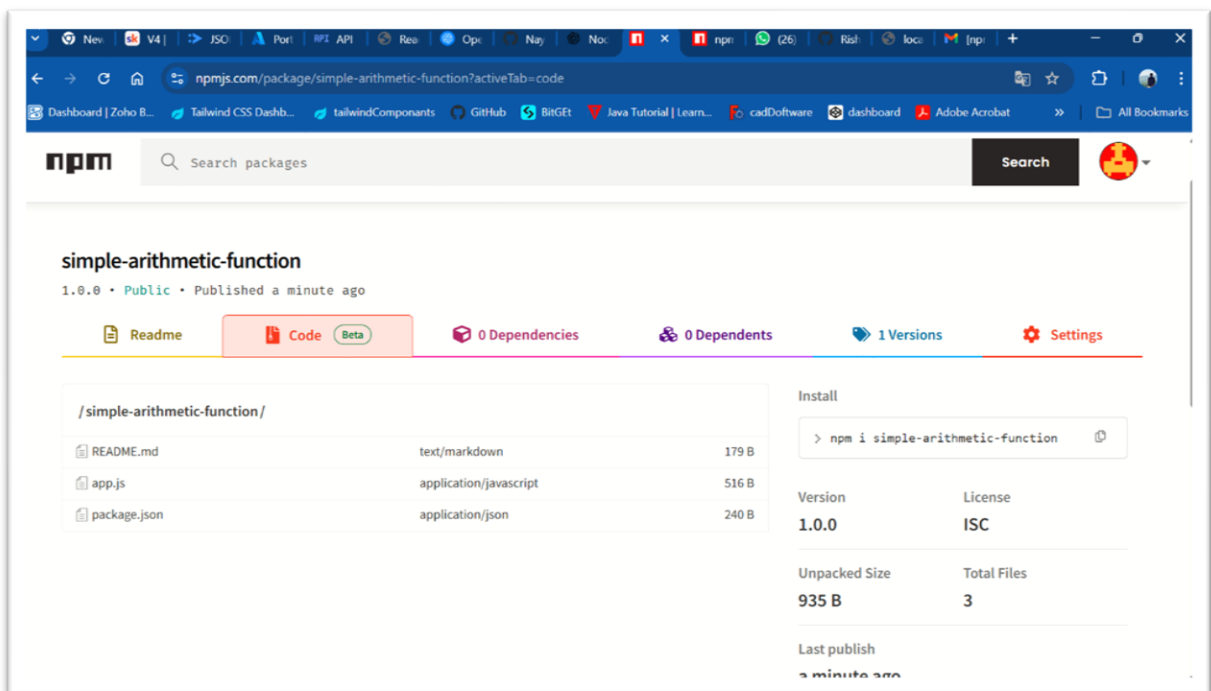


The screenshot shows a VS Code terminal window with the following output:

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Current file name: D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q10\app.js
Process Platform: win32
Memory Use: {
  rss: 31232000,
  heapTotal: 5046272,
  heapUsed: 4099432,
  external: 450658,
  arrayBuffers: 17678
}
Executed after events using setImmediate
○ Executed after 1 second
PS D:\MSCICT\Sem-3\NodeJS\Practical-Assignment-1\35_Nayan_Padhiyar_304\Q10>
```

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

Output :-



[GitHub Link:

https://github.com/Nayan8319/OSWD/tree/main/35_Nayan_Padhiyar_304_A1]