

NAME : Drashti Sureshbhai Desai

ROLL NO: 09

SUB : Open Source Web Development

STD: ICT 3rd

QUTION: 1

- **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.)**

AJAX call

`Call /gethello`

hello nodejs...

2. Develop a web server which serves static resources.



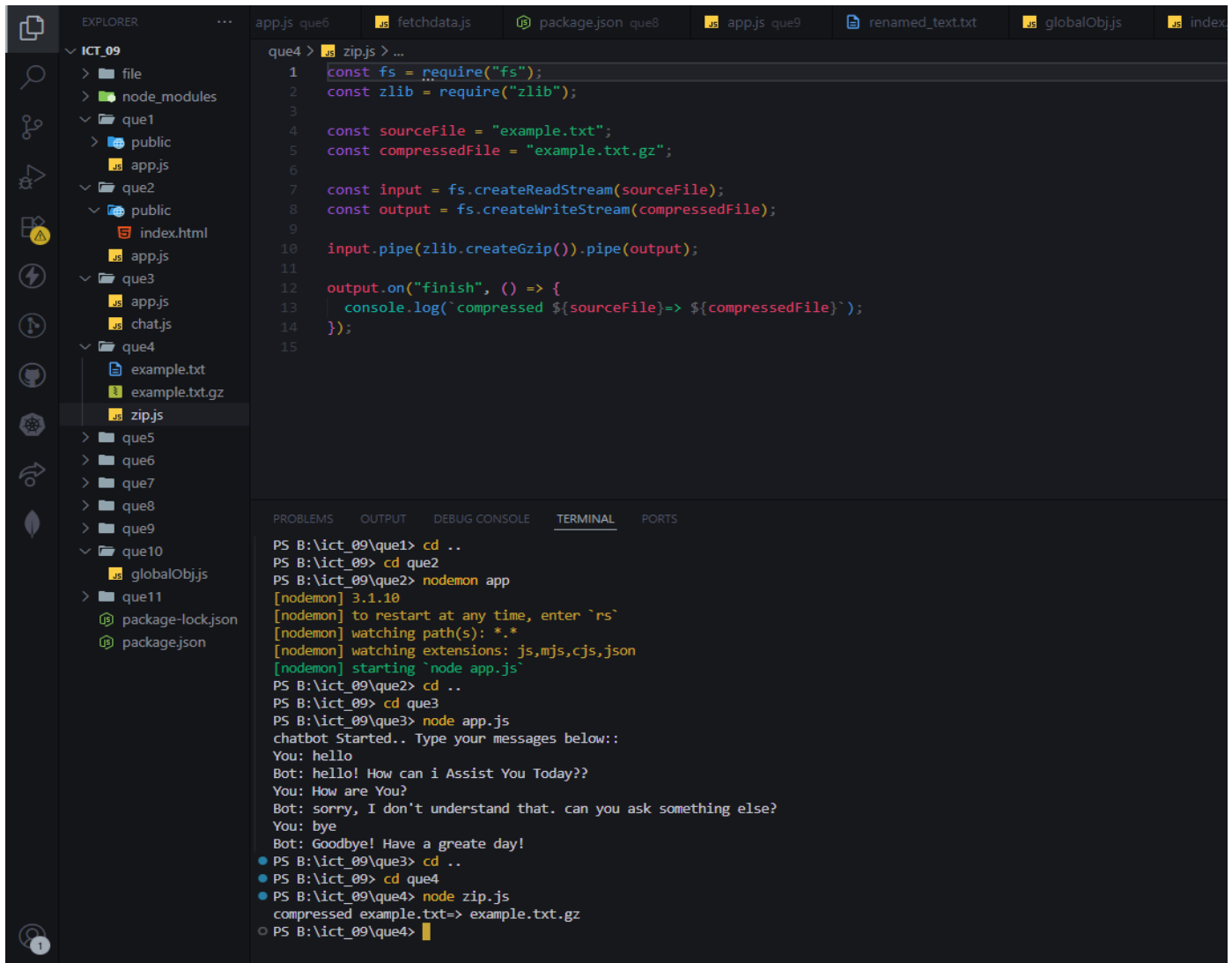
Welcome to the static server!!!

This is a HTML Page....

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

```
PS B:\ict_09\que3> node app.js
chatbot Started.. Type your messages below::
You: hello
Bot: hello! How can i Assist You Today??
You: How are You?
Bot: sorry, I don't understand that. can you ask something else?
You: bye
Bot: Goodbye! Have a greate day!
```

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

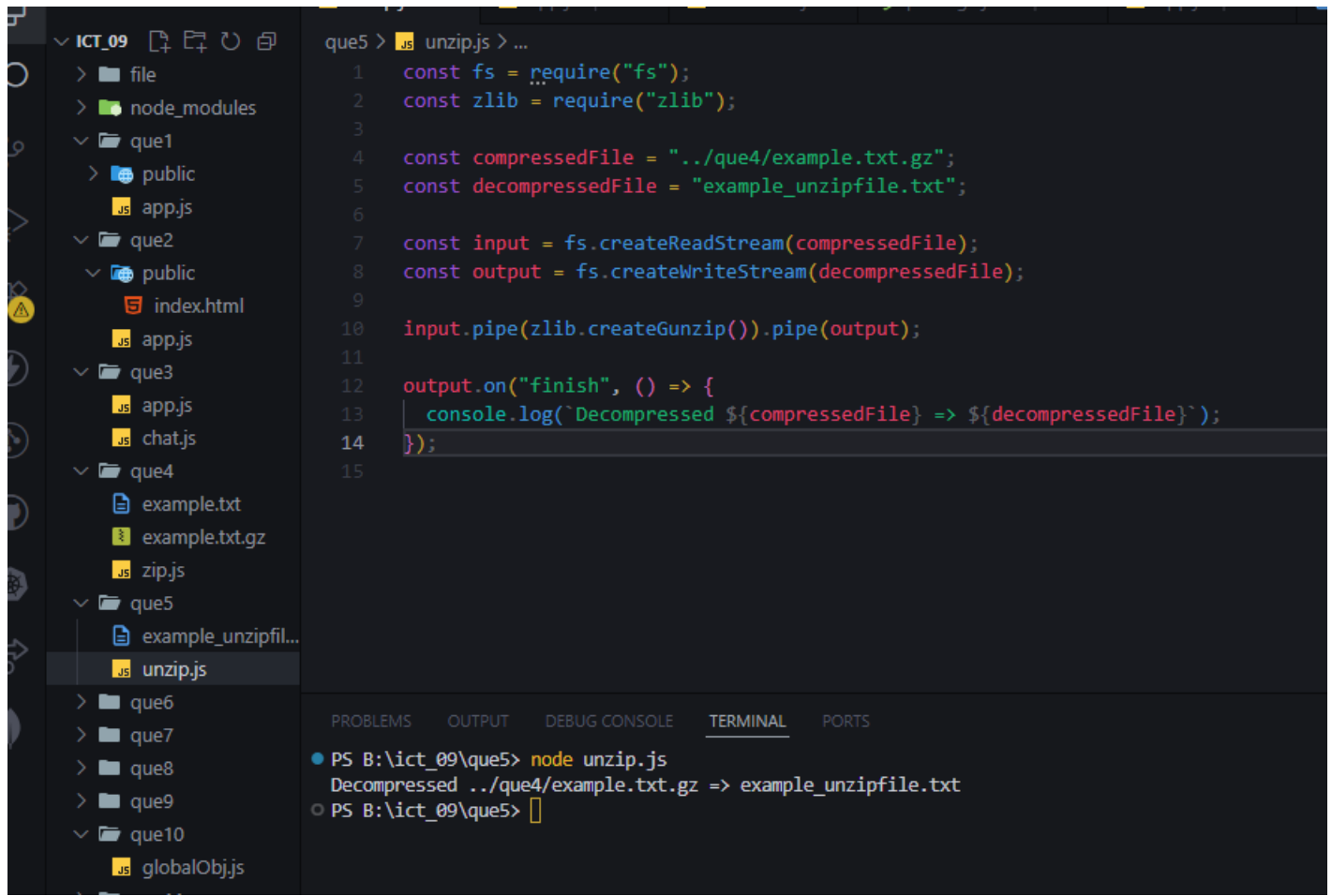


The screenshot displays the Visual Studio Code interface. The Explorer sidebar on the left shows a project structure with folders 'que1' through 'que11'. The file 'zip.js' is selected in the 'que4' folder. The main editor shows the code for 'zip.js', which uses the 'fs' and 'zlib' modules to compress 'example.txt' into 'example.txt.gz'. The bottom panel shows the 'TERMINAL' tab with a series of commands and their outputs, demonstrating the workflow from directory navigation to file compression.

```
1 const fs = require("fs");
2 const zlib = require("zlib");
3
4 const sourceFile = "example.txt";
5 const compressedFile = "example.txt.gz";
6
7 const input = fs.createReadStream(sourceFile);
8 const output = fs.createWriteStream(compressedFile);
9
10 input.pipe(zlib.createGzip()).pipe(output);
11
12 output.on("finish", () => {
13   console.log(`compressed ${sourceFile}=> ${compressedFile}`);
14 });
15
```

PS B:\ict_09\que1> cd ..
PS B:\ict_09> cd que2
PS B:\ict_09\que2> nodemon app
[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node app.js`
PS B:\ict_09\que2> cd ..
PS B:\ict_09> cd que3
PS B:\ict_09\que3> node app.js
chatbot Started.. Type your messages below::
You: hello
Bot: hello! How can i Assist You Today??
You: How are You?
Bot: sorry, I don't understand that. can you ask something else?
You: bye
Bot: Goodbye! Have a greate day!
● PS B:\ict_09\que3> cd ..
● PS B:\ict_09> cd que4
● PS B:\ict_09\que4> node zip.js
compressed example.txt=> example.txt.gz
○ PS B:\ict_09\que4>

5. Write a program to extract a zip file.



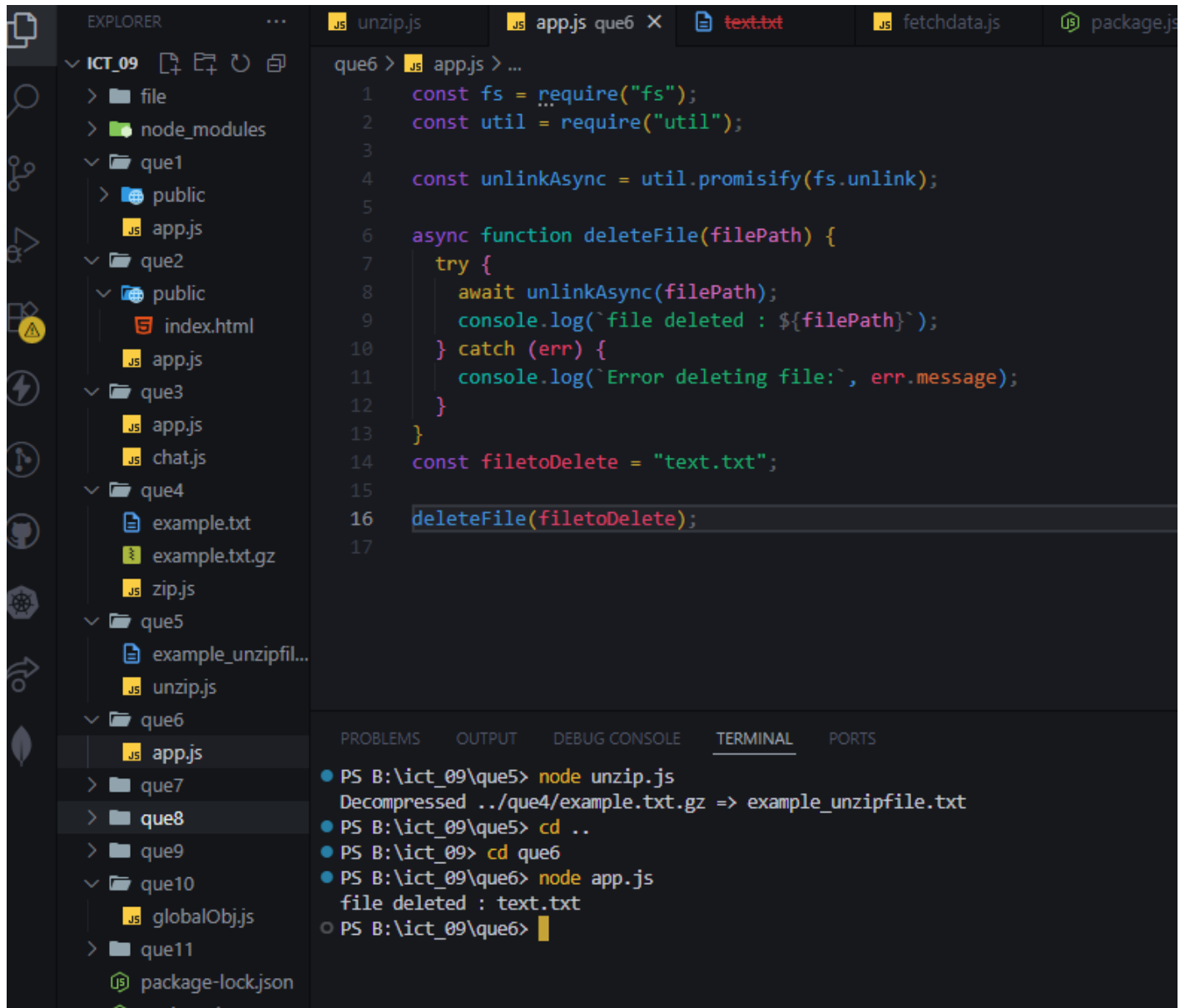
The screenshot shows a VS Code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders 'que1' through 'que10' and files like 'app.js', 'index.html', 'example.txt', 'example.txt.gz', and 'zip.js'. The code editor shows the content of 'unzip.js' in the 'que5' folder. The code uses the 'fs' and 'zlib' modules to read a gzipped file and write its decompressed content to a new file. The terminal at the bottom shows the command 'node unzip.js' being executed, resulting in the message 'Decompressed ../que4/example.txt.gz => example_unzipfile.txt'.

```
que5 > .js unzip.js > ...
1  const fs = require("fs");
2  const zlib = require("zlib");
3
4  const compressedFile = "../que4/example.txt.gz";
5  const decompressedFile = "example_unzipfile.txt";
6
7  const input = fs.createReadStream(compressedFile);
8  const output = fs.createWriteStream(decompressedFile);
9
10 input.pipe(zlib.createGunzip()).pipe(output);
11
12 output.on("finish", () => {
13   console.log(`Decompressed ${compressedFile} => ${decompressedFile}`);
14 });
15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS B:\ict_09\que5> node unzip.js
Decompressed ../que4/example.txt.gz => example_unzipfile.txt
- PS B:\ict_09\que5>

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



The image shows a Visual Studio Code editor with a project structure on the left and a code editor on the right. The project structure includes folders for 'ICT_09' and 'que6', with various files like 'app.js', 'chat.js', 'example.txt', 'example.txt.gz', 'zip.js', 'example_unzipfil...', 'unzip.js', 'globalObj.js', and 'package-lock.json'. The code editor displays the following JavaScript code in 'app.js':

```
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(`file deleted : ${filePath}`);
10   } catch (err) {
11     console.log(`Error deleting file:`, err.message);
12   }
13 }
14
15 const fileToDelete = "text.txt";
16 deleteFile(fileToDelete);
17
```

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

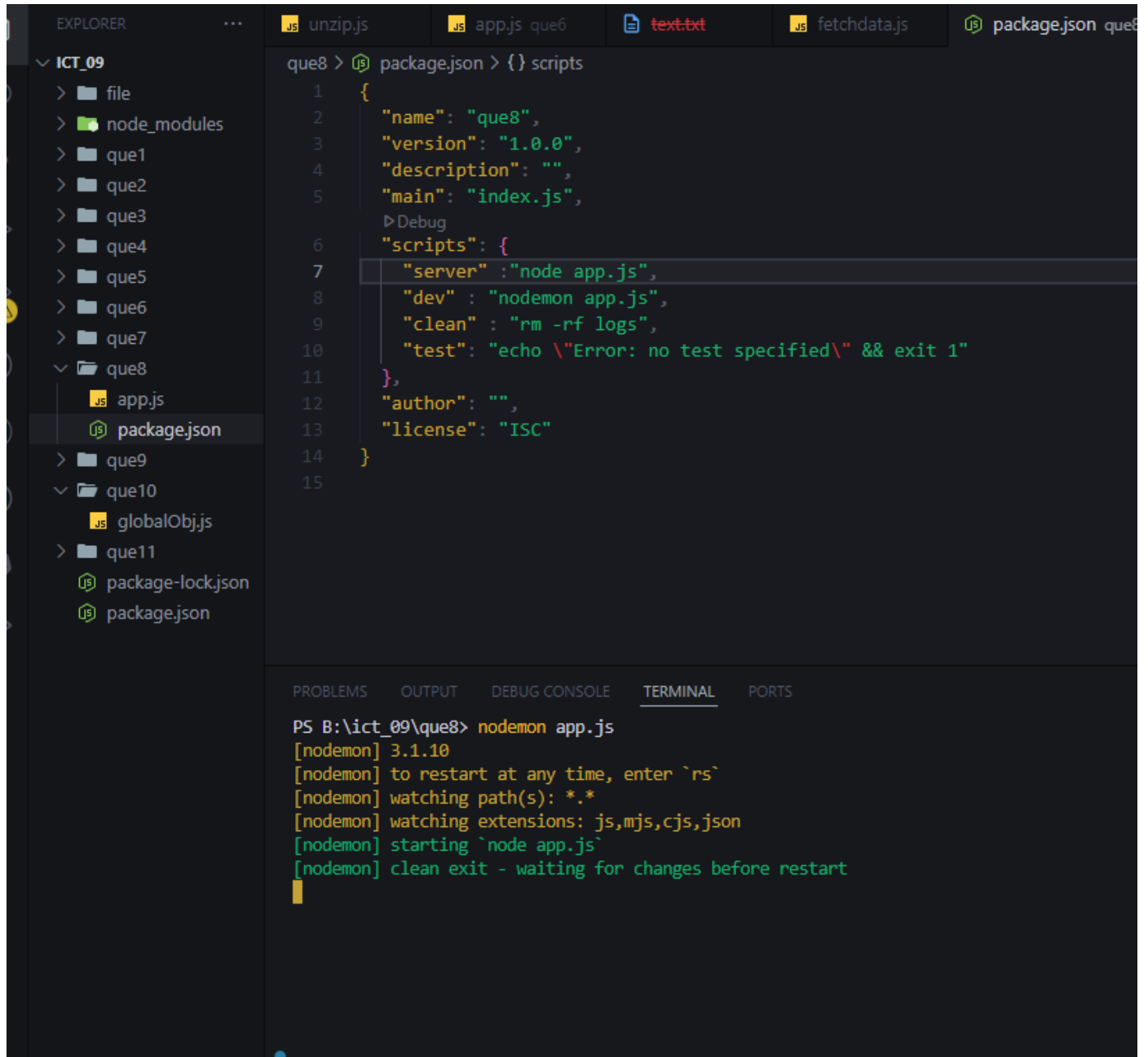
```
PS B:\ict_09\que5> node unzip.js
Decompressed ../que4/example.txt.gz => example_unzipfile.txt
PS B:\ict_09\que5> cd ..
PS B:\ict_09> cd que6
PS B:\ict_09\que6> node app.js
file deleted : text.txt
PS B:\ict_09\que6>
```

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

```
PS B:\ict_09\que7> node fetchdata.js
Google page Html 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/googleg/1x/googleg_standard_color_128dp.png" item
prop="image"><title>Google</title><script nonce="87c8NIjoponZG3_Fa-FCw">(function(){var _g={kEI:'mk6IaNNtBtFNisQPx3T8As',kEXP1:'0,202854,2,3497404,689,435,447880,90781,14111,64702,6398,211574,142929,247320,42724,11106,5230576,36812642,2
5306698,39200,20961,1411
PS B:\ict_09\que7>
```

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



The screenshot shows a Visual Studio Code editor with the Explorer sidebar on the left displaying a project structure. The main editor area shows the `package.json` file for a project named `que8`. The file contains the following JSON:

```
{
  "name": "que8",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "server": "node app.js",
    "dev": "nodemon app.js",
    "clean": "rm -rf logs",
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

The terminal at the bottom shows the command `nodemon app.js` being executed, with the following output:

```
PS B:\ict_09\que8> nodemon app.js
[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node app.js`
[nodemon] clean exit - waiting for changes before restart
```

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

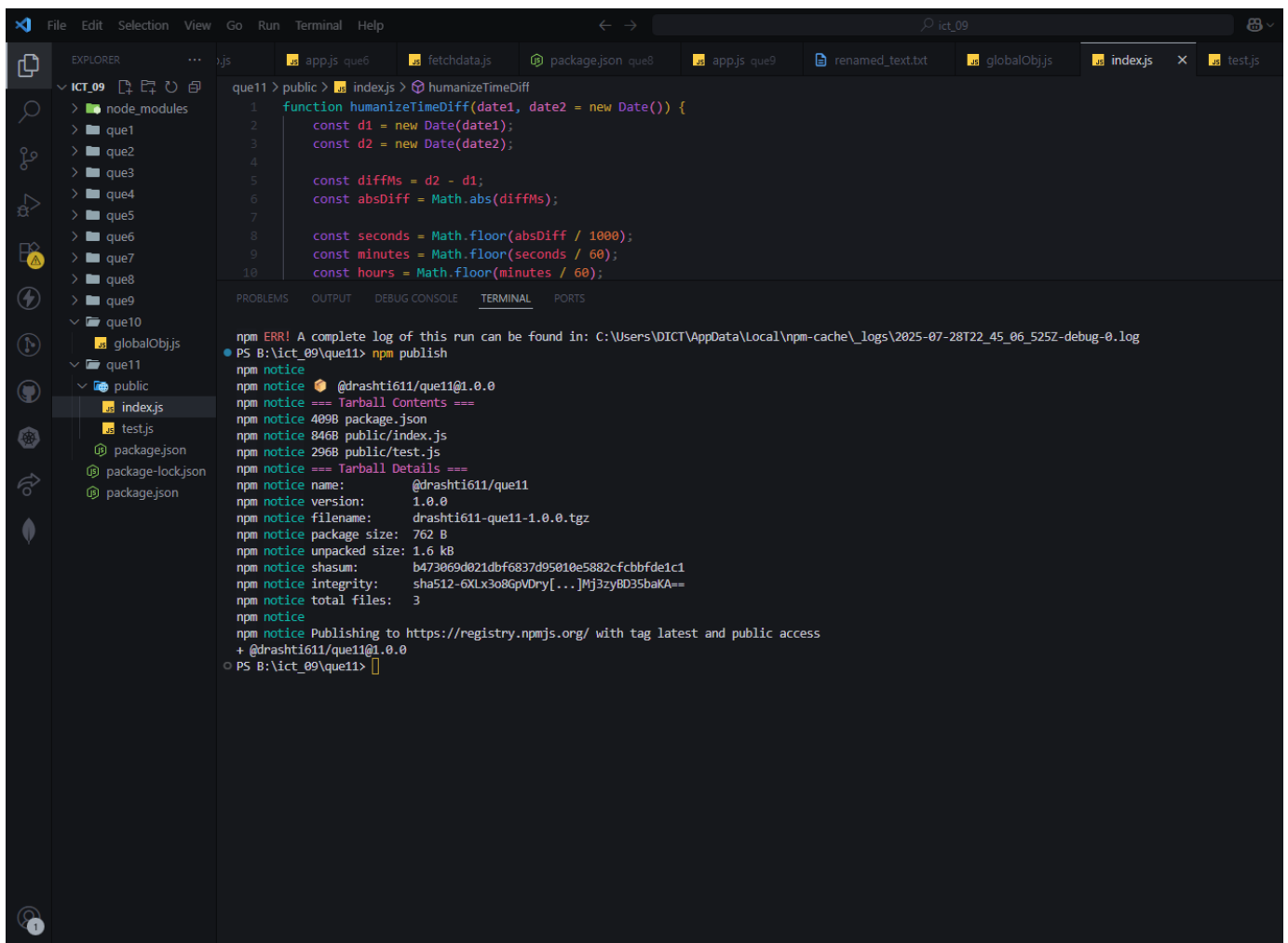
```
PS B:\ict_09\que9> nodemon app.js
[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
• [nodemon] starting `node app.js`
file written successfully..
appending data successfully...
file content:
  hello,this is a same file..
• Appending more text.

file renamed...
[nodemon] clean exit - waiting for changes before restart
```


10. A program which uses global objects in nodejs.

```
● PS B:\ict_09\que10> node globalObj.js
Current File Path: B:\ict_09\que10\globalObj.js
Current Directory Path: B:\ict_09\que10
Node.js Version: v18.18.2
Platform: win32
Home Directory: C:\Users\DICT
Accessing Global Variable: This is a global variable
Logging from the global console object.
Exiting program in 3 seconds...
This message is shown after 2 seconds delay using setTimeout.
Exiting now.
○ PS B:\ict_09\que10>
```

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



The screenshot shows the Visual Studio Code interface with a project named 'que11' open. The Explorer sidebar on the left shows the file structure, including a 'public' directory with 'index.js' and 'test.js'. The main editor displays the code for 'index.js', which defines a 'humanizeTimeDiff' function. The Output window at the bottom shows the terminal output of the 'npm publish' command, including package details and the successful upload to the npm registry.

```
que11 > public > indexjs > humanizeTimeDiff
1 function humanizeTimeDiff(date1, date2 = new Date()) {
2   const d1 = new Date(date1);
3   const d2 = new Date(date2);
4
5   const diffMs = d2 - d1;
6   const absDiff = Math.abs(diffMs);
7
8   const seconds = Math.floor(absDiff / 1000);
9   const minutes = Math.floor(seconds / 60);
10  const hours = Math.floor(minutes / 60);
11
12  return { seconds, minutes, hours };
13 }
14
15 module.exports = humanizeTimeDiff;
```

```
npm ERR! A complete log of this run can be found in: C:\Users\DICT\AppData\Local\npm-cache\_logs\2025-07-28T22_45_06_525Z-debug-0.log
● PS B:\ict_09\que11> npm publish
npm notice
npm notice 📦 @drashti611/que11@1.0.0
npm notice === Tarball Contents ===
npm notice 4098 package.json
npm notice 8468 public/index.js
npm notice 2968 public/test.js
npm notice === Tarball Details ===
npm notice name: @drashti611/que11
npm notice version: 1.0.0
npm notice filename: drashti611-que11-1.0.0.tgz
npm notice package size: 762 B
npm notice unpacked size: 1.6 kB
npm notice shasum: b473069d021dbf6837d95010e5882cfcbfde1c1
npm notice integrity: sha512-6XLx3o8GpVDry[...]Mj3zyBD35baKA==
npm notice total files: 3
npm notice
npm notice Publishing to https://registry.npmjs.org/ with tag latest and public access
+ @drashti611/que11@1.0.0
○ PS B:\ict_09\que11>
```

