Practical Assignment – 1

Name : Sakshi Umeshkumar Maisuria

Semester : 7th sem (MSc-IT)

Roll No. : 026

Subject : Application Development using Full Stack

Code : 705

1. Develop a web server with following functionalities:

- Serve static resources.
- Handle GET request.
- Handle POST request.

Code:

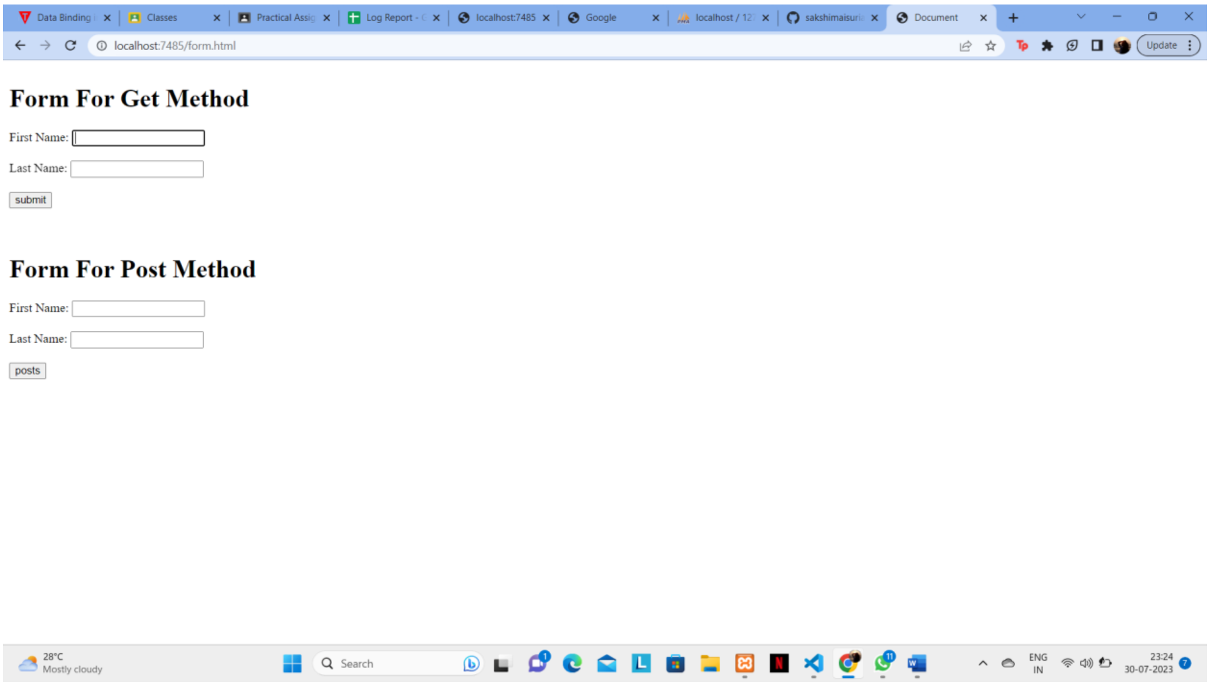
q1.js:

```
const http = require("http");

const url = require("url");
const nstatic = require("node-static");
const { readSync } = require("fs");
const fs = new nstatic.Server("./files");

var server = http.createServer((req, res) => {
  // res.end("p1");
  const url1 = url.parse(req.url, true);
  if (req.url == "form") {
    fs.readFile("./files/form.html", (err, data) => {
      if (err) {
        res.writeHead(404, { "Content-Type": "text/html" });
        res.write("404:file not found");
      } else {
        res.writeHead(200, { "Content-Type": "text/html" });
        res.write(data);
        res.end();
      }
    });
  } else if (url1.pathname == "/form_get" && req.method == "GET") {
    res.write(
      "Your First Name:" +
        url1.query.fname +
        " And Your Last Name:" +
        url1.query.lname
    );
    res.end();
  } else if (url1.pathname == "/form_post" && req.method == "POST") {
    var body = "";
    req.on("data", (postdata) => {
      body += postdata.toString();
    });
    req.on("end", function () {
      res.write(body);
      res.end();
    });
  }
  req
    .addListener("end", function () {
      fs.serve(req, res);
    })
    .resume();
});
server.listen(7485, () => {
  console.log("port 7485");
});
```

form.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <div style="display: flex;">
        <div>
            <h1>Form For Get Method</h1>
            <form action="/form_get" method="get">
                First Name: <input type="text" id="fname"
name="fname"><br><br>
                Last Name: <input type="text" id="lname" name="lname"><br><br>
                <input type="submit" value="submit" name="submit">
            </form>
            <br><br>
            <h1>Form For Post Method</h1>
            <form action="/form_post" method="post">
                First Name: <input type="text" id="pfname"
name="pfname"><br><br>
                Last Name: <input type="text" id="plname"
name="plname"><br><br>
                <input type="submit" value="posts" name="posts">
            </form>
        </div>

    </div>
</body>
</html>
```

Output:



localhost:7485/form.html

Form For Get Method

First Name:

Last Name:

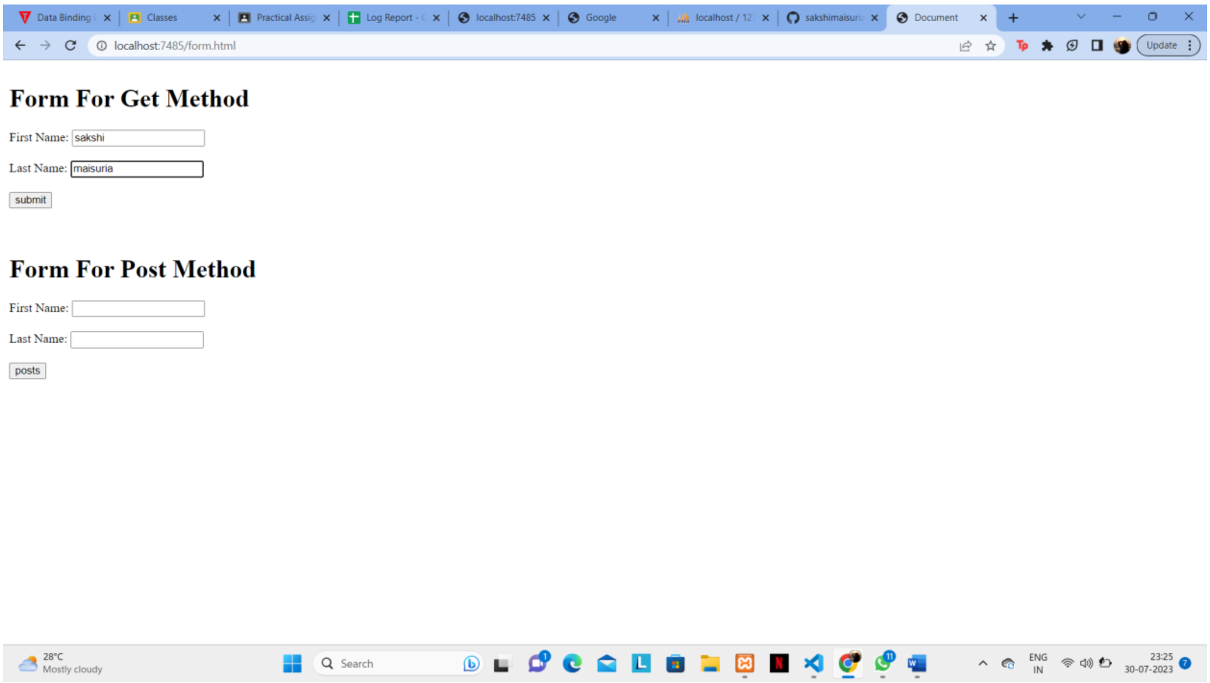
submit

Form For Post Method

First Name:

Last Name:

posts



localhost:7485/form.html

Form For Get Method

First Name: sakshi

Last Name: maisuria

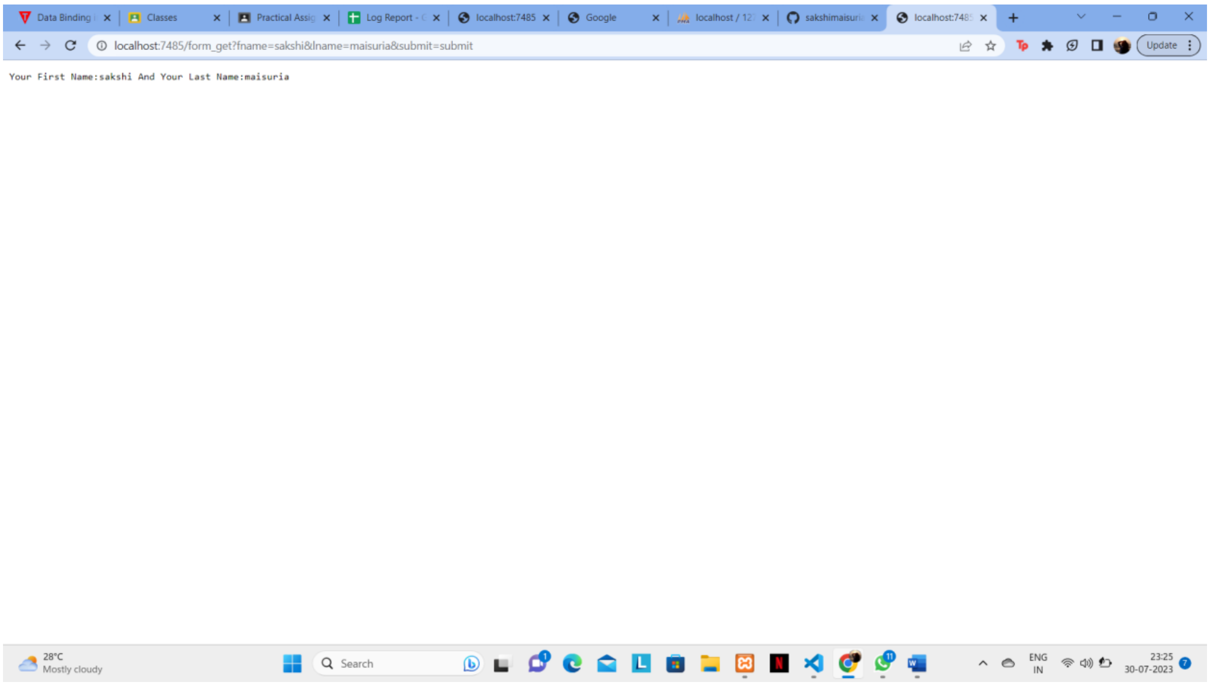
submit

Form For Post Method

First Name:

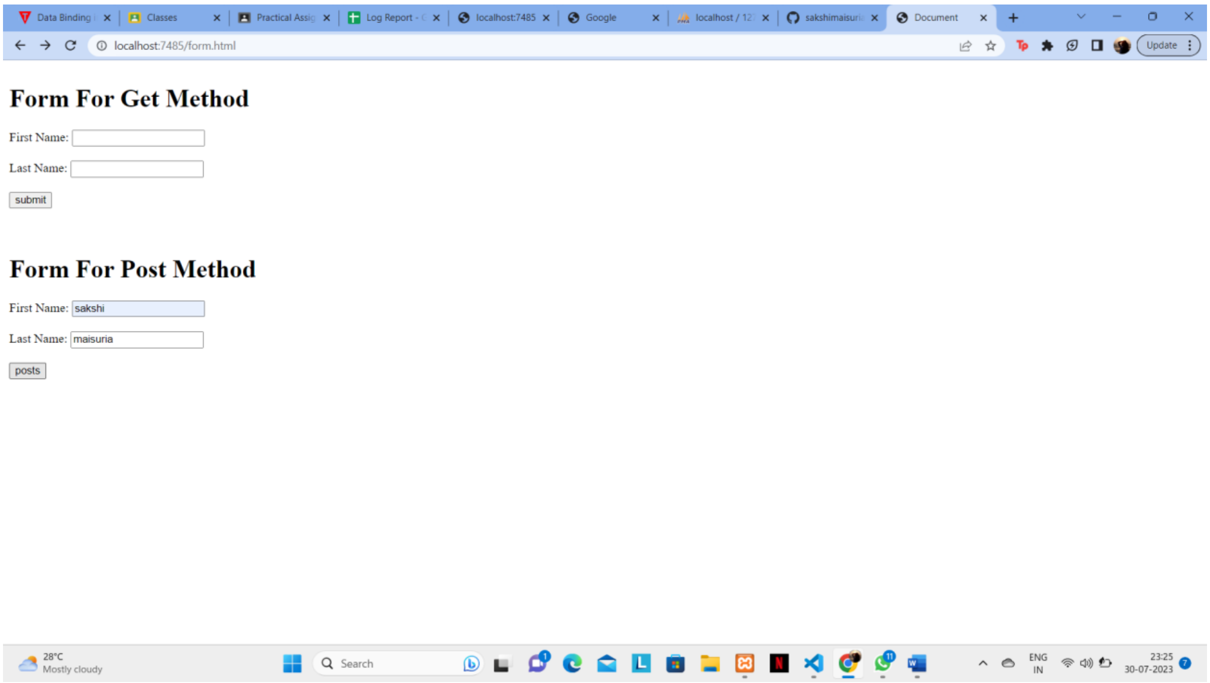
Last Name:

posts



localhost:7485/form_get?fname=sakshi&lname=maisuria&submit=submit

```
Your First Name:sakshi And Your Last Name:maisuria
```



localhost:7485/form.html

Form For Get Method

First Name:

Last Name:

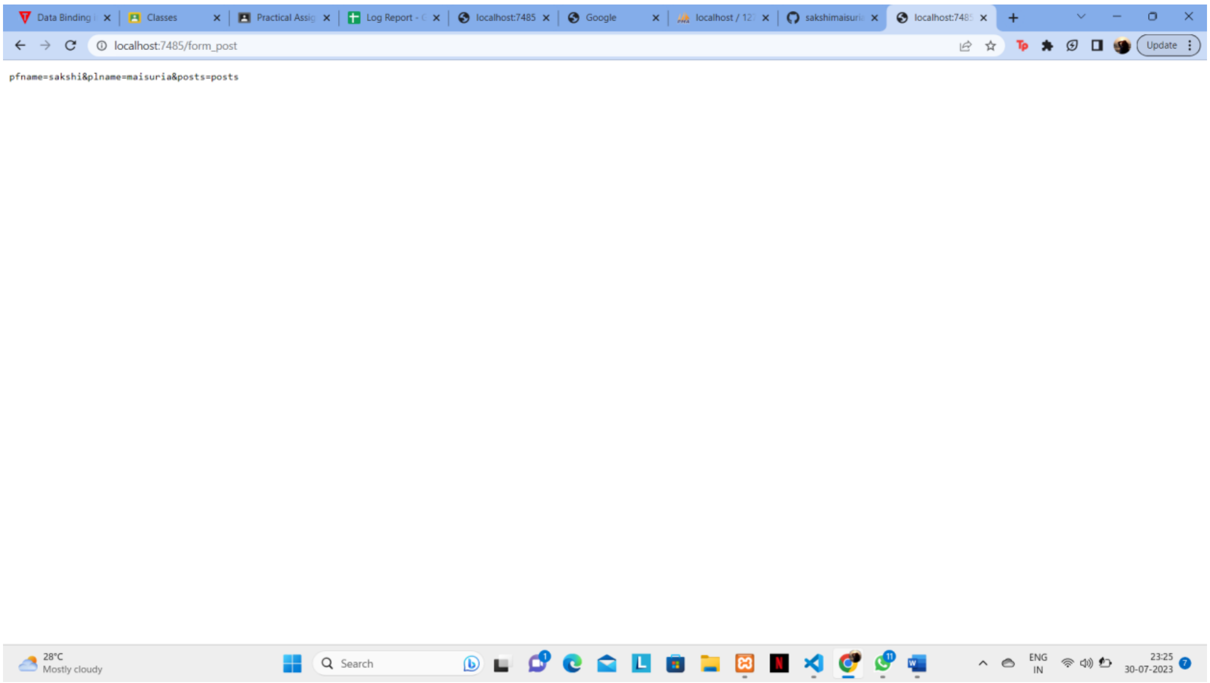
submit

Form For Post Method

First Name: sakshi

Last Name: maisuria

posts



```
pfname=sakshi&plname=maisuria&posts=posts
```

2. 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.)

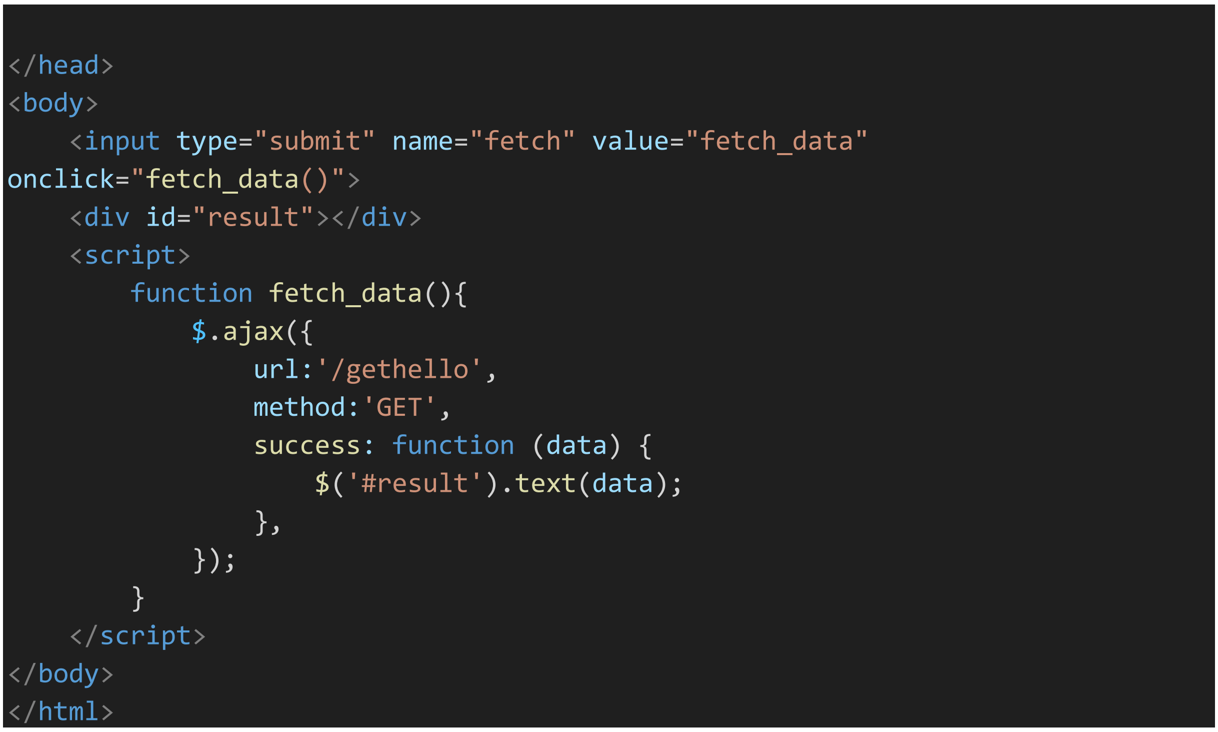
Code:

q2.js:

```
const http = require("http");
const fs = require("fs");

const server = http
  .createServer((req, res) => {
    if (req.url === "/gethello" && req.method === "GET") {
      res.writeHead(200, { "Content-Type": "Text/html" });
      res.write("Hello NodeJS!!!");
      res.end();
    } else if (req.url === "/" && req.method === "GET") {
      res.writeHead(500, { "Content-Type": "Text/html" });
      res.write("internal server error!!");
      res.end();
    }

    fs.readFile("./files/q2.html", null, (err, data) => {
      if (err) {
        res.writeHead(404);
        res.write("file not found");
      } else {
        res.write(data);
      }
      res.end();
    });
  })
  .listen(7485);
```

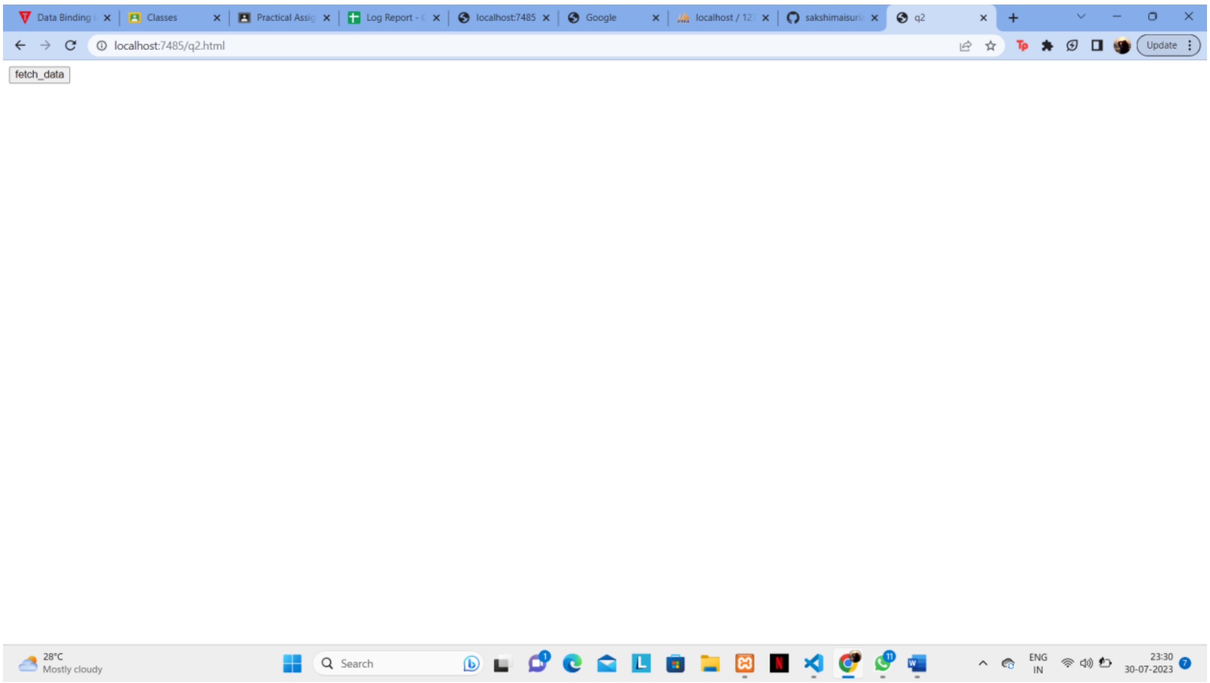
q2.html:

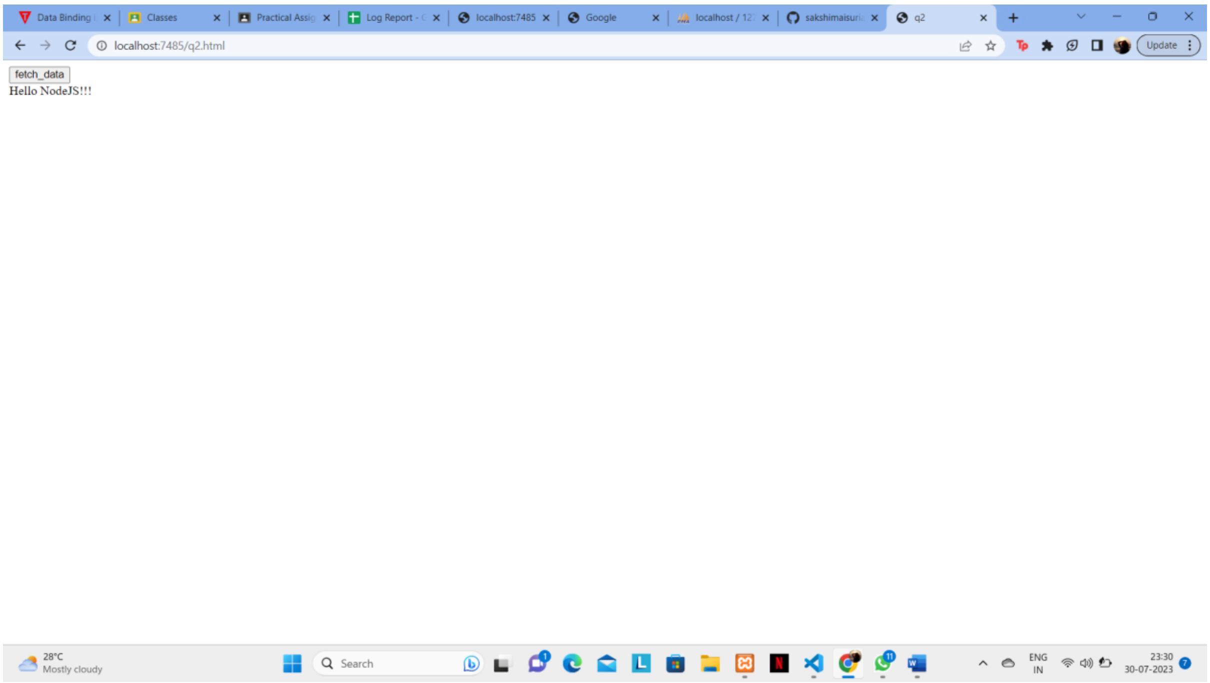
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>q2</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```



```
</head>
<body>
    <input type="submit" name="fetch" value="fetch_data"
onclick="fetch_data()">
    <div id="result"></div>
    <script>
        function fetch_data(){
            $.ajax({
                url:'/gethello',
                method:'GET',
                success: function (data) {
                    $('#result').text(data);
                },
            });
        }
    </script>
</body>
</html>
```

Output:





fetch_data

Hello NodeJS!!!

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

Code:

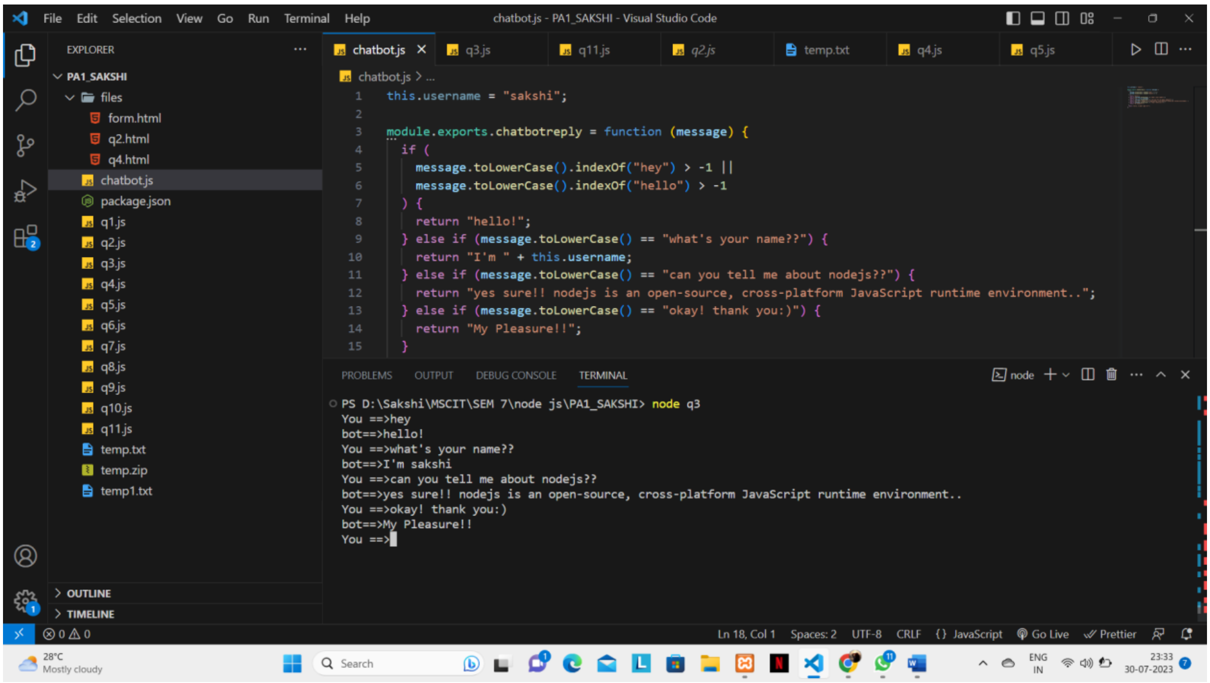
q3.js:

```
var readline = require("readline"); //user input
var chatbot = require("./chatbot");

var interface = readline.createInterface(process.stdin, process.stdout);
//craete interface for input output
interface.setPrompt("You ==>"); //create prompt
interface.prompt();

interface
  .on("line", (message) => {
    console.log("bot==>" + chatbot.chatbotreply(message));

    interface.prompt();
  })
  .on("close", () => {
    process.exit(0);
  });
```

Chatbot.js:

```
this.username = "sakshi";

module.exports.chatbotreply = function (message) {
  if (
    message.toLowerCase().indexOf("hey") > -1 ||
    message.toLowerCase().indexOf("hello") > -1
  ) {
    return "hello!";
  } else if (message.toLowerCase() == "what's your name??") {
    return "I'm " + this.username;
  } else if (message.toLowerCase() == "can you tell me about nodejs??") {
    return "yes sure!! nodejs is an open-source, cross-platform JavaScript
runtime environment..";
  } else if (message.toLowerCase() == "okay! thank you:)") {
    return "My Pleasure!!";
  }
  return "sorry, I didn't get it!!";
};
```

Output:



```javascript
this.username = "sakshi";

module.exports.chatbotreply = function (message) {
  if (
    message.toLowerCase().indexOf("hey") > -1 ||
    message.toLowerCase().indexOf("hello") > -1
  ) {
    return "hello!";
  } else if (message.toLowerCase() == "what's your name??") {
    return "I'm " + this.username;
  } else if (message.toLowerCase() == "can you tell me about nodejs??") {
    return "yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..";
  } else if (message.toLowerCase() == "okay! thank you:)") {
    return "My Pleasure!!";
  }
```

```
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q3
You ==>hey
bot==>hello!
You ==>what's your name??
bot==>I'm sakshi
You ==>can you tell me about nodejs??
bot==>yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
You ==>okay! thank you:)
bot==>My Pleasure!!
You ==>
```

4. Use above chatbot module in web based chatting of websocket.

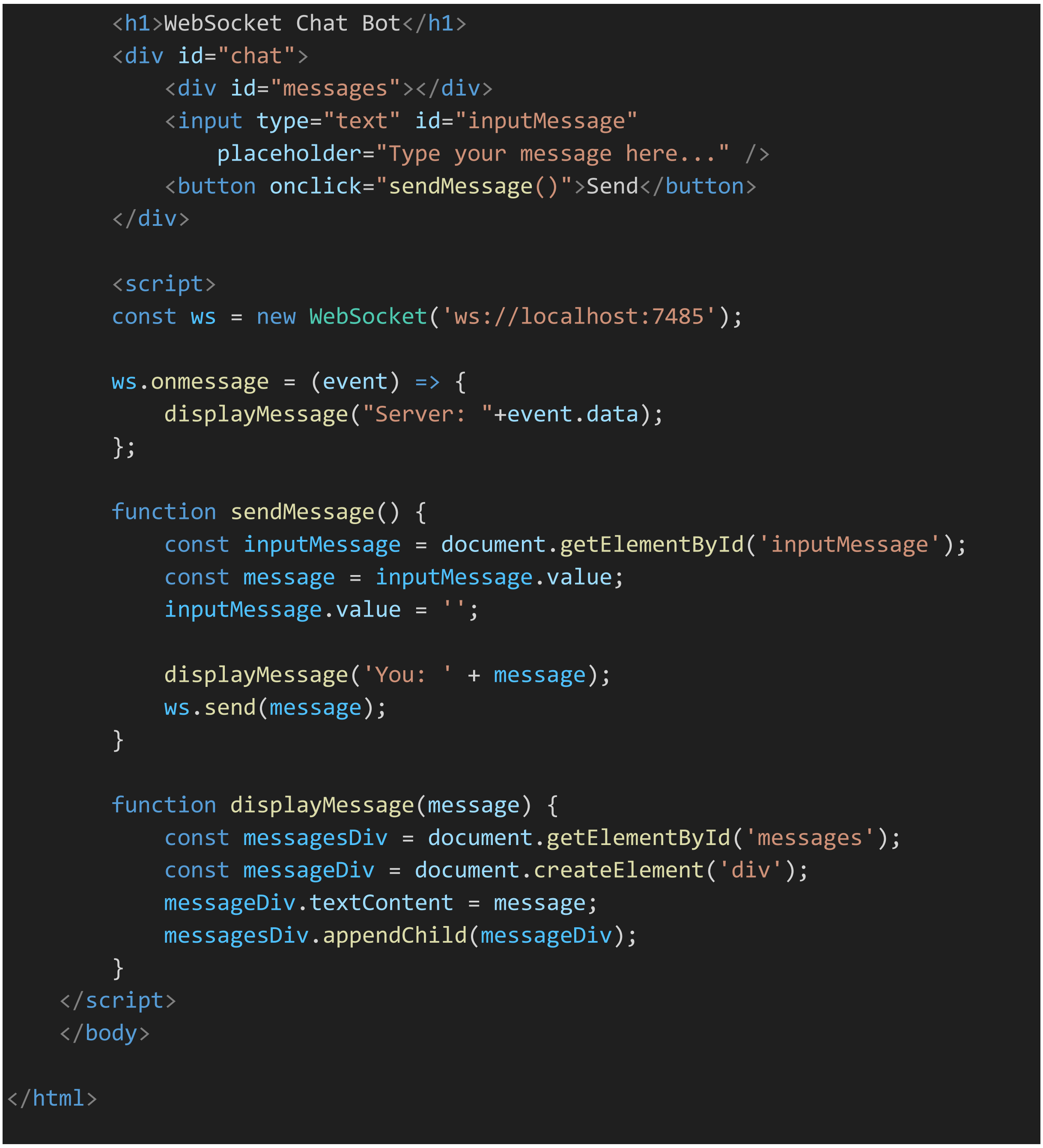
Code:

q4.js:

```
const http = require("http");
const ns = require("node-static");
const chatbot = require("./chatbot"); //import chatbot.js module
const WebSocket = require("ws");
var url = require("url");

const file = new ns.Server("./files/q4.html");

const server = http.createServer((req, res) => {
  req
    .on("end", () => {
      file.serve(req, res);
    })
    .resume();
});

server.listen(7485, () => {
  console.log("Server listening on 7485");
});

const wss = new WebSocket.Server({ server: server });

wss.on("connection", (ws) => {
  ws.send("Hii, I am a chatbot!!");

  ws.on("message", (data) => {
    const message = data.toString();
    const reply = chatbot.chatbotreply(message);
    console.log(reply);
    ws.send(reply);
  });
});
```

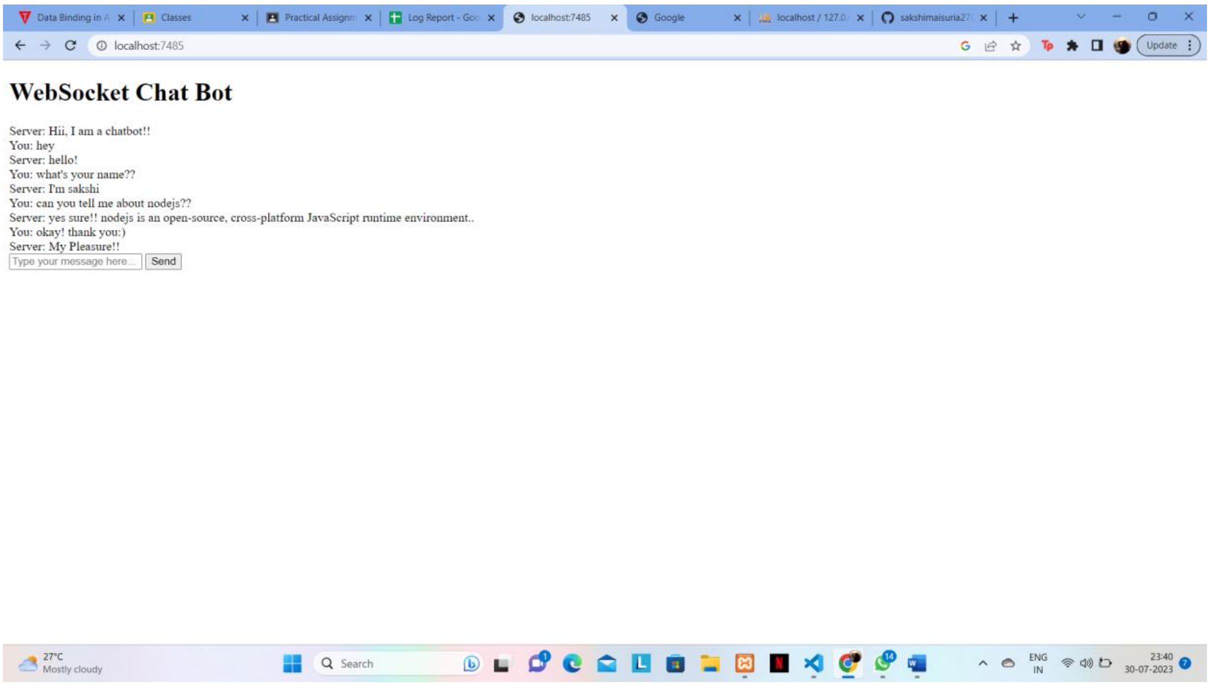
q4.html:

```
<!DOCTYPE html>
<html>

    <head>
        <meta charset="utf-8">
        <title></title>
    </head>

    <body>
```



```
        <h1>WebSocket Chat Bot</h1>
        <div id="chat">
            <div id="messages"></div>
            <input type="text" id="inputMessage"
                placeholder="Type your message here..." />
            <button onclick="sendMessage()">Send</button>
        </div>

        <script>
        const ws = new WebSocket('ws://localhost:7485');

        ws.onmessage = (event) => {
            displayMessage("Server: "+event.data);
        };

        function sendMessage() {
            const inputMessage = document.getElementById('inputMessage');
            const message = inputMessage.value;
            inputMessage.value = '';

            displayMessage('You: ' + message);
            ws.send(message);
        }

        function displayMessage(message) {
            const messagesDiv = document.getElementById('messages');
            const messageDiv = document.createElement('div');
            messageDiv.textContent = message;
            messagesDiv.appendChild(messageDiv);
        }
    </script>
    </body>

</html>
```

Output:



WebSocket Chat Bot

```
Server: Hii, I am a chatbot!!
You: hey
Server: hello!
You: what's your name??
Server: I'm sakshi
You: can you tell me about nodejs??
Server: yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
You: okay! thank you:)
Server: My Pleasure!!
```

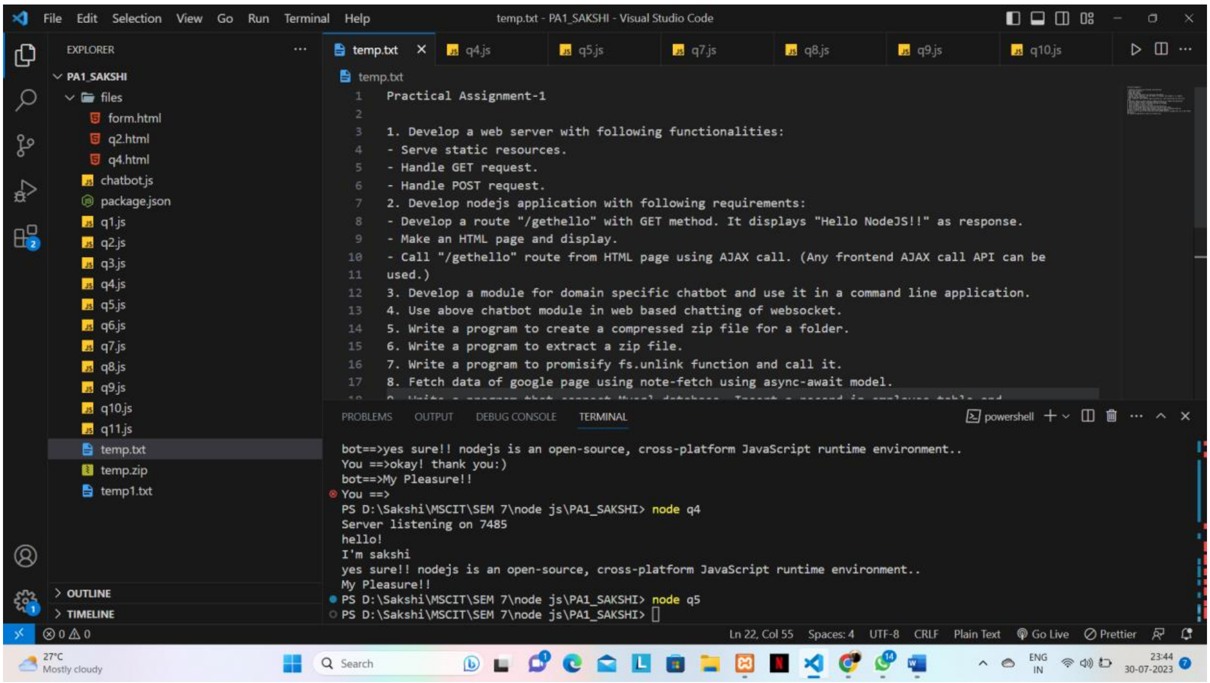
5. Write a program to create a compressed zip file for a folder.

Code:

q5.js:

```
const zlib = require("zlib");
const fs = require("fs");

const inputFile = fs.createReadStream("temp.txt");
const outputFile = fs.createWriteStream("temp.zip");

inputFile.pipe(zlib.createGzip()).pipe(outputFile);
```

Output:



```
Practical Assignment-1

1. Develop a web server with following functionalities:
- Serve static resources.
- Handle GET request.
- Handle POST request.
2. 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.)
3. Develop a module for domain specific chatbot and use it in a command line application.
4. Use above chatbot module in web based chatting of websocket.
5. Write a program to create a compressed zip file for a folder.
6. Write a program to extract a zip file.
7. Write a program to promisify fs.unlink function and call it.
8. Fetch data of google page using note-fetch using async-await model.
```

```
bot==>yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
You ==>okay! thank you:)
bot==>My Pleasure!!
You ==>
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q4
Server listening on 7485
hello!
I'm sakshi
yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
My Pleasure!!
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q5
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI>
```

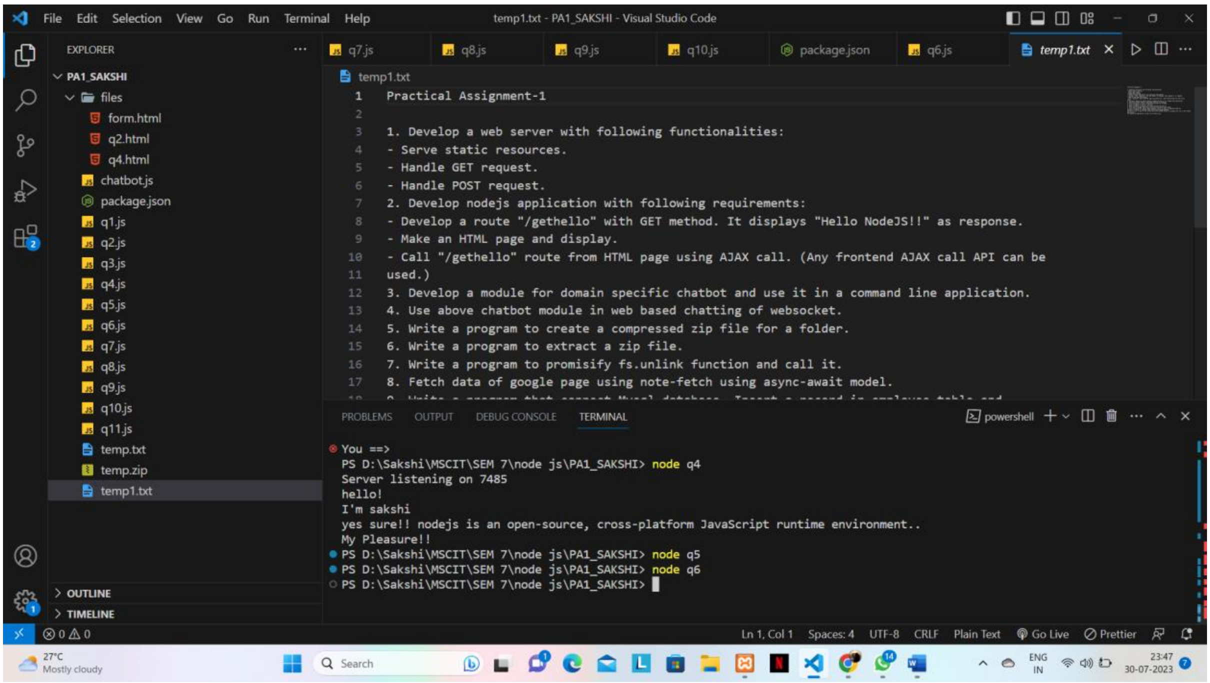
6. Write a program to extract a zip file.

Code:

q6.js:

```
const zlib = require("zlib");
const fs = require("fs");

const inputFile = fs.createReadStream("temp.zip");
const outputFile = fs.createWriteStream("temp1.txt");

inputFile.pipe(zlib.createUnzip()).pipe(outputFile);
```

Output:



```
Practical Assignment-1

1. Develop a web server with following functionalities:
- Serve static resources.
- Handle GET request.
- Handle POST request.
2. 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.)
3. Develop a module for domain specific chatbot and use it in a command line application.
4. Use above chatbot module in web based chatting of websocket.
5. Write a program to create a compressed zip file for a folder.
6. Write a program to extract a zip file.
7. Write a program to promisify fs.unlink function and call it.
8. Fetch data of google page using note-fetch using async-await model.
9. Write a program that connect Mysql database. Insert a record in employee table and
```

```
You ==>
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q4
Server listening on 7485
hello!
I'm sakshi
yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
My Pleasure!!
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q5
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q6
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI>
```

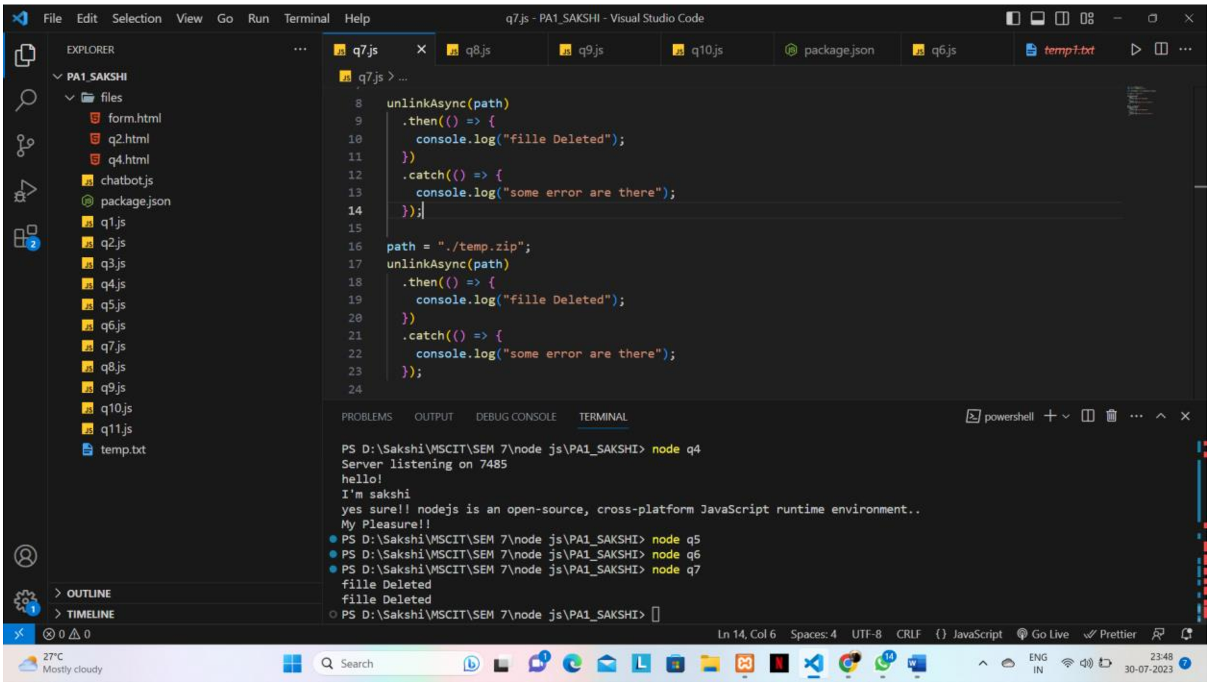
7. Write a program to promisify fs.unlink function and call it.

Code:

q7.js:

```
const fs = require("fs");
const util = require("util");

const unlinkAsync = util.promisify(fs.unlink);

var path = "./temp1.txt";

unlinkAsync(path)
  .then(() => {
    console.log("fille Deleted");
  })
  .catch(() => {
    console.log("some error are there");
  });

path = "./temp.zip";
unlinkAsync(path)
  .then(() => {
    console.log("fille Deleted");
  })
  .catch(() => {
    console.log("some error are there");
  });
```

Output:



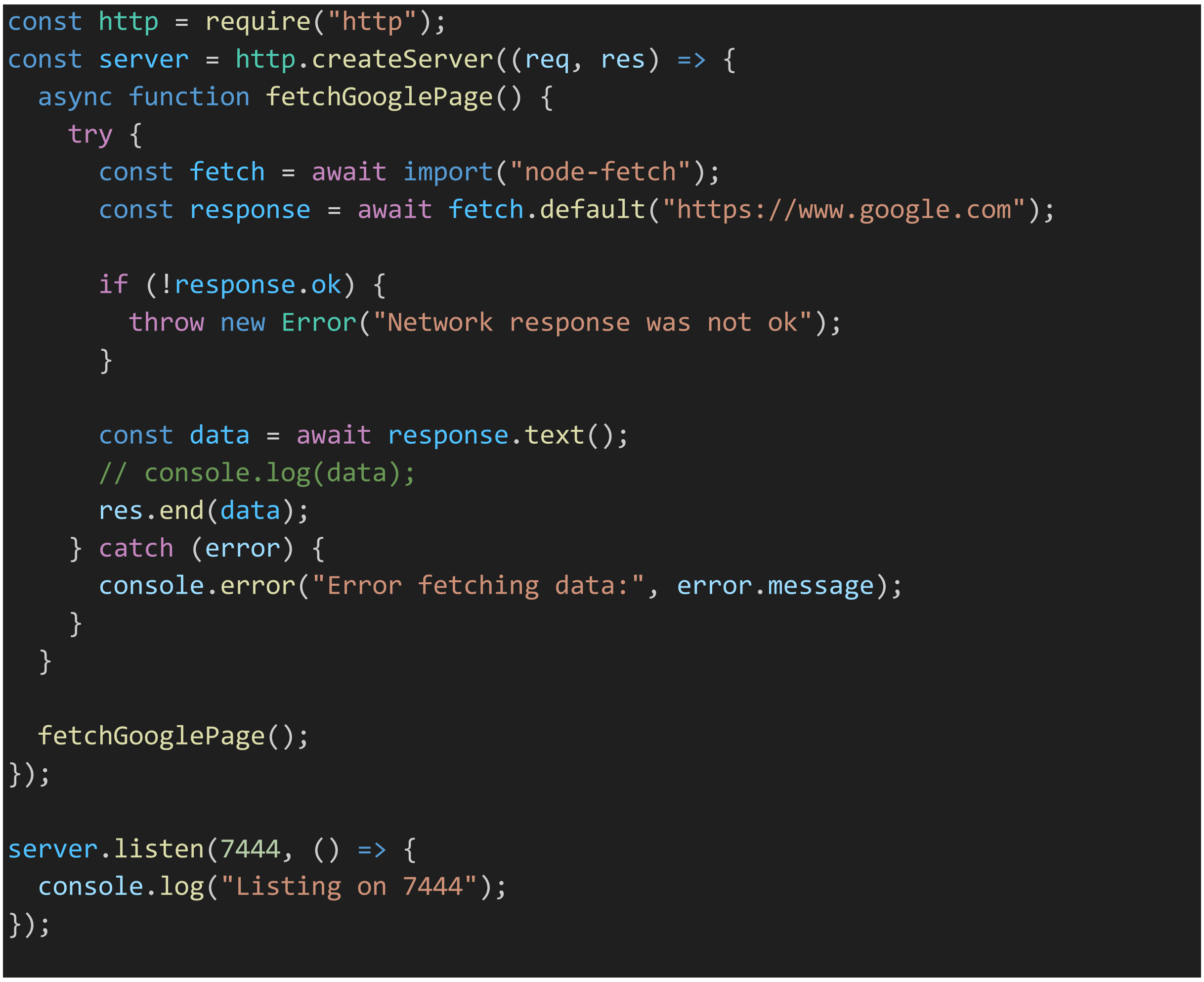
```javascript
unlinkAsync(path)
  .then(() => {
    console.log("fille Deleted");
  })
  .catch(() => {
    console.log("some error are there");
  });

path = "./temp.zip";
unlinkAsync(path)
  .then(() => {
    console.log("fille Deleted");
  })
  .catch(() => {
    console.log("some error are there");
  });
```

```
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q4
Server listening on 7485
hello!
I'm sakshi
yes sure!! nodejs is an open-source, cross-platform JavaScript runtime environment..
My Pleasure!!
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q5
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q6
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q7
fille Deleted
fille Deleted
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI>
```

8. Fetch data of google page using note-fetch using async-await model.

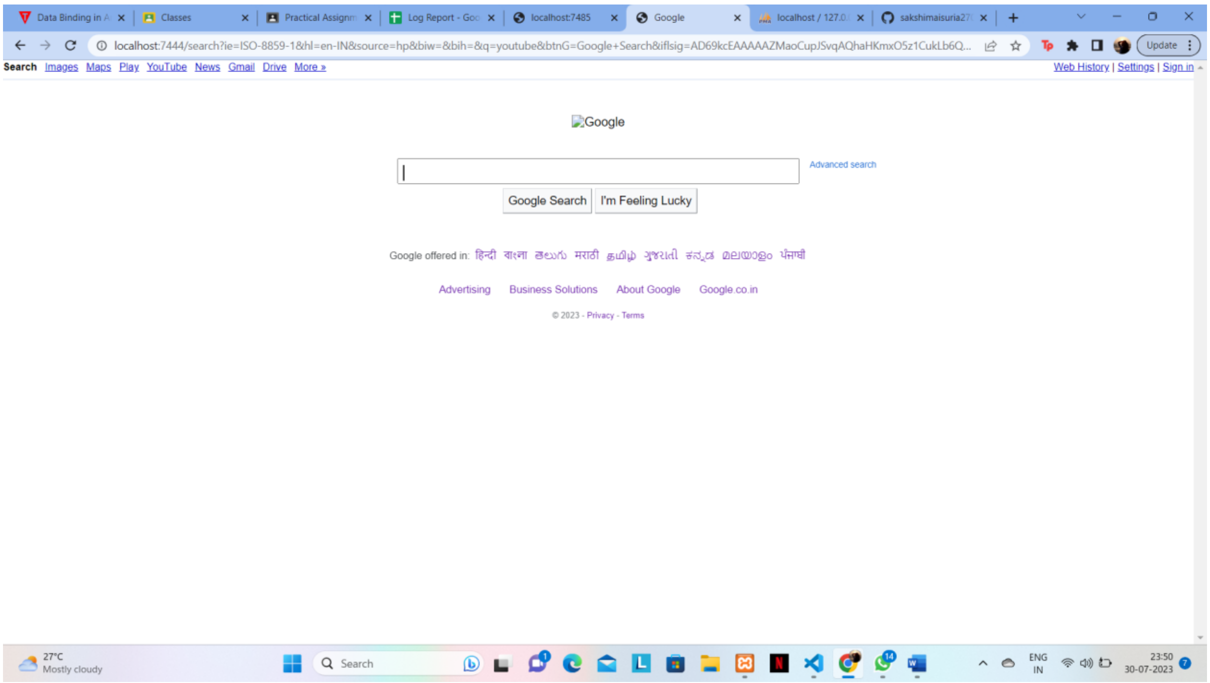
Code:

q8.js:



```
const http = require("http");
const server = http.createServer((req, res) => {
  async function fetchGooglePage() {
    try {
      const fetch = await import("node-fetch");
      const response = await fetch.default("https://www.google.com");

      if (!response.ok) {
        throw new Error("Network response was not ok");
      }

      const data = await response.text();
      // console.log(data);
      res.end(data);
    } catch (error) {
      console.error("Error fetching data:", error.message);
    }
  }

  fetchGooglePage();
});

server.listen(7444, () => {
  console.log("Listing on 7444");
});
```

Output:



localhost:7444/search?ie=ISO-8859-1&hl=en-IN&source=hp&biw=&bih=&q=youtube&btnG=Google+Search&iflsig=AD69kcEAAAAAZMaoCupJSvqAQhaHKmxO5z1CukLb6Q...

Google offered in: हिन्दी বাংলা తెలుగు मराठी தமிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ

© 2023 - Privacy - Terms

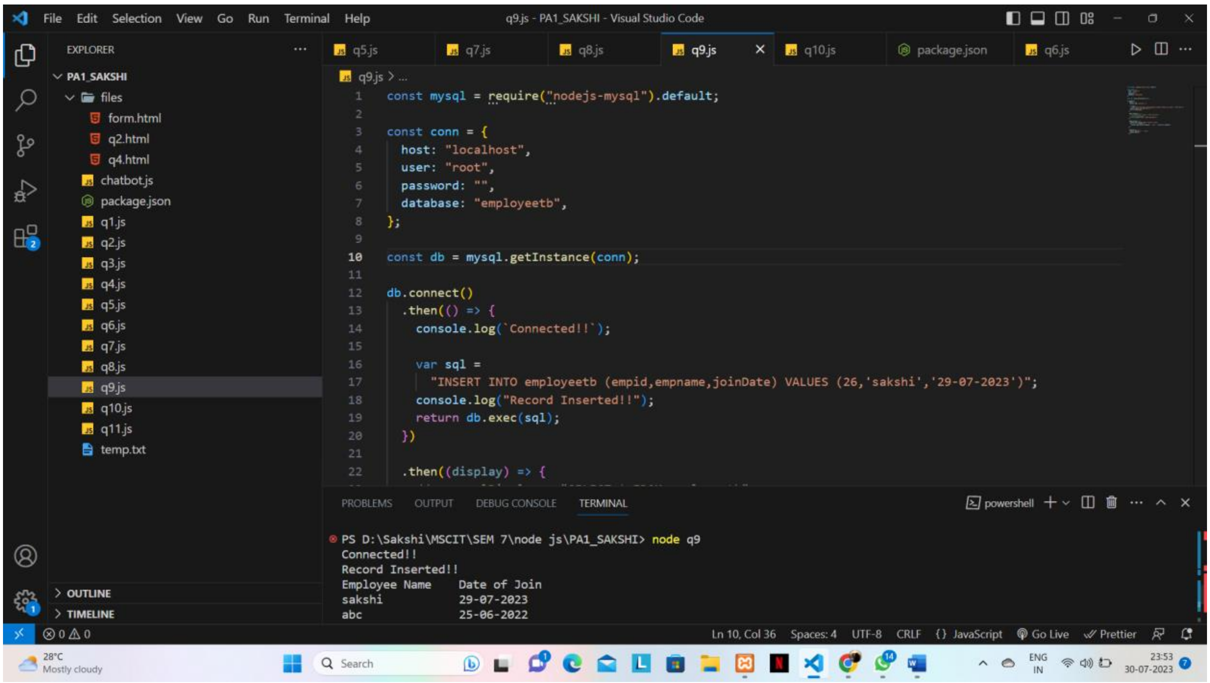
9. Write a program that connect Mysql database, Insert a record in employee table and display all records in employee table using promise based approach.

Code:

q9.js:

```
const mysql = require("nodejs-mysql").default;

const conn = {
  host: "localhost",
  user: "root",
  password: "",
  database: "employeetb",
};

const db = mysql.getInstance(conn);

db.connect()
  .then(() => {
    console.log(`Connected!!`);

    var sql =
      "INSERT INTO employeetb (empid,empname,joinDate) VALUES
(26,'sakshi','29-07-2023')";
    console.log("Record Inserted!!");
    return db.exec(sql);
  })

  .then((display) => {
    // var sqlDisplay = "SELECT * FROM employeetb";
    // console.log(display);
    return db.exec("SELECT * FROM employeetb");
  })

  .then((result) => {
    console.log("Employee Name \t Date of Join");
    for (var i in result) {
      console.log(result[i].empname + " \t\t " + result[i].joinDate);
    }
  })

  .catch((err) => {
    console.log("Error: " + err);
    process.exit(0);
  });
```

Output:

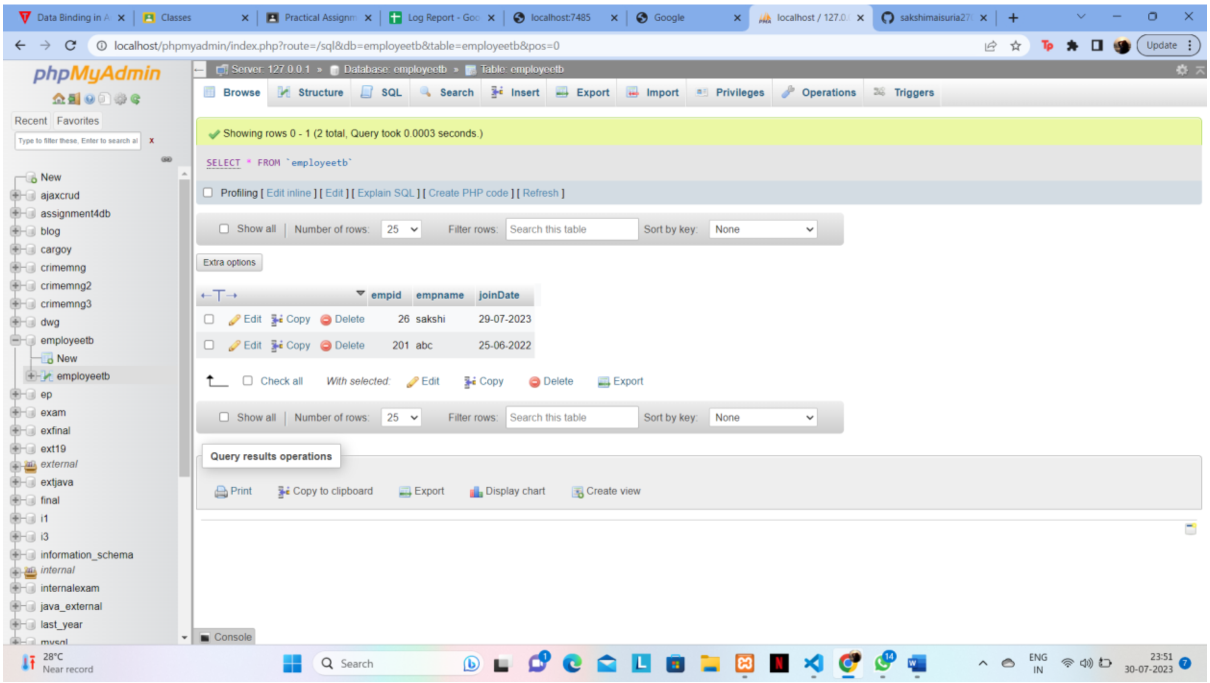


q9.js - PA1_SAKSHI - Visual Studio Code

```
const mysql = require("nodejs-mysql").default;

const conn = {
  host: "localhost",
  user: "root",
  password: "",
  database: "employeetb",
};

const db = mysql.getInstance(conn);

db.connect()
  .then(() => {
    console.log(`Connected!!`);

    var sql =
      "INSERT INTO employeetb (empid,empname,joinDate) VALUES (26,'sakshi','29-07-2023')";
    console.log("Record Inserted!!");
    return db.exec(sql);
  })

  .then((display) => {
```

```
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q9
Connected!!
Record Inserted!!
Employee Name     Date of Join
sakshi            29-07-2023
abc               25-06-2022
```

Mysql db:



localhost/phpmyadmin/index.php?route=/sql&db=employeetb&table=employeetb&pos=0

Server: 127.0.0.1 » Database: employeetb » Table: employeetb

✓ Showing rows 0 - 1 (2 total, Query took 0.0003 seconds.)

```
SELECT * FROM `employeetb`
```

				empid	empname	joinDate
□	Edit	Copy	Delete	26	sakshi	29-07-2023
□	Edit	Copy	Delete	201	abc	25-06-2022

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

Code:

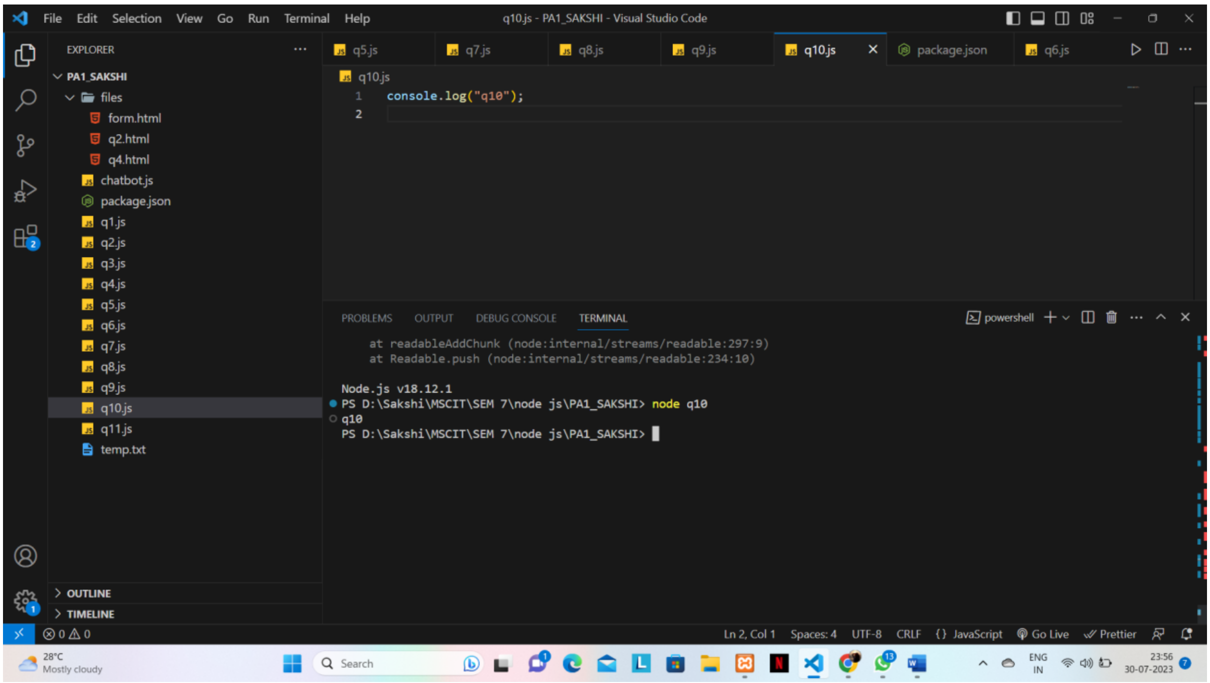
package.json:

```
{
  "name": "pa1_sakshi",
  "version": "1.0.0",
  "description": "",
  "main": "chatbot.js",
  "scripts": {
    "start":"node q10.js",
    "test": "echo \"Error: no test specified\" && exit 1",
    "script1": "node q10.js",
    "script2": "node q2.js",
    "script3": "node q3.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

q10.js:

```
console.log("q10");
```

output:

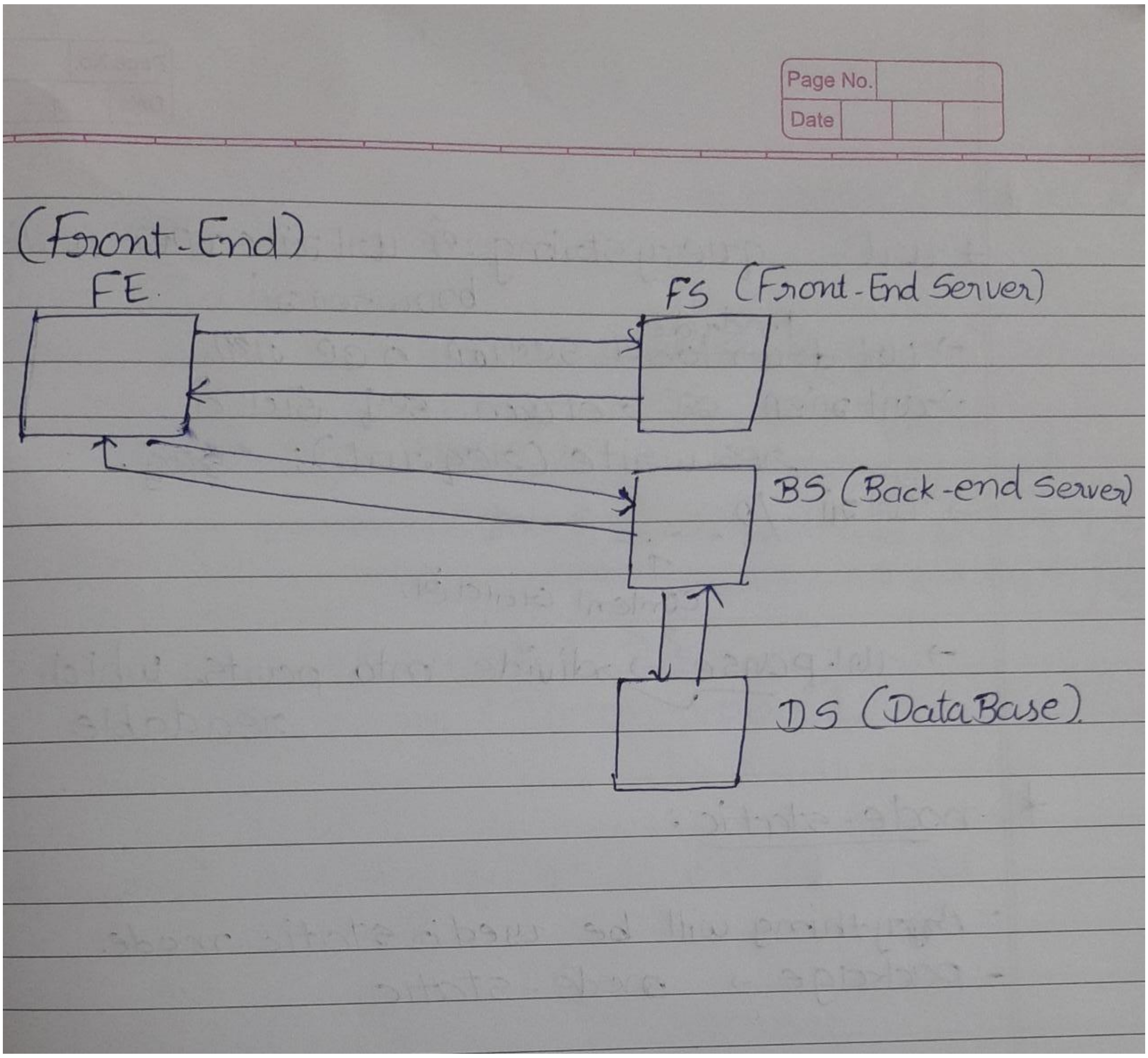


q10.js - PA1_SAKSHI - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
PA1_SAKSHI
files
form.html
q2.html
q4.html
chatbot.js
package.json
q1.js
q2.js
q3.js
q4.js
q5.js
q6.js
q7.js
q8.js
q9.js
q10.js
q11.js
temp.txt
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell
Ln 2, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript Go Live Prettier

```
console.log("q10");
```

```
    at readableAddChunk (node:internal/streams/readable:297:9)
    at Readable.push (node:internal/streams/readable:234:10)

Node.js v18.12.1
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI> node q10
q10
PS D:\Sakshi\MSCIT\SEM 7\node js\PA1_SAKSHI>
```

11. Develop an application to show live cricket score.



Page No.
Date
(Front-End)
FE.
FS (Front-End Server)
BS (Back-end Server)
DS (DataBase).