NAME : GOSAI BIPASHA RAJUBHAI

ROLL NO. : 18

SEM :- 7TH

SUBJECT : FULL STACK -705

ASSIGNMNET :- 1

1. Develop a web server with following functionalities:

- Serve static resources.

- Handle GET request.

- Handle POST request.

Server.js

```
const http = require('http');
const fs = require('fs');
const path = require('path');
const url = require('url');
const querystring = require('querystring');

const PORT = 3000;

// Function to serve static files
const serveStaticFile = (res, filePath, contentType) => {
    fs.readFile(filePath, (error, content) => {
        if (error) {
            res.writeHead(500);
            res.end(`Sorry, there was an error: ${error.code} ..\n`);
        } else {
            res.writeHead(200, { 'Content-Type': contentType });
            res.end(content, 'utf-8');
        }
    });
};

// Create the server
```

```
const server = http.createServer((req, res) => {
  const parsedUrl = url.parse(req.url, true);

  // Handle GET request
  if (req.method === 'GET') {
    if (parsedUrl.pathname === '/') {
      serveStaticFile(res, path.join(__dirname, 'public', 'index.html'), 'text/html');
    } else if (parsedUrl.pathname === '/submit') {
      // Handle form submission here if needed
      res.writeHead(200, { 'Content-Type': 'text/plain' });
      res.end('Form submitted successfully!');
    } else {
      // Serve other static files
      const filePath = path.join(__dirname, 'public', parsedUrl.pathname);
      const extname = String(path.extname(filePath)).toLowerCase();
      const mimeTypes = {
        '.html': 'text/html',
        '.js': 'text/javascript',
        '.css': 'text/css',
        '.json': 'application/json',
        '.png': 'image/png',
        '.jpg': 'image/jpg',
        '.gif': 'image/gif',
        '.svg': 'image/svg+xml',
        '.wav': 'audio/wav',
        '.mp4': 'video/mp4',
        '.woff': 'application/font-woff',
        '.ttf': 'application/font-ttf',
        '.eot': 'application/vnd.ms-fontobject',
```

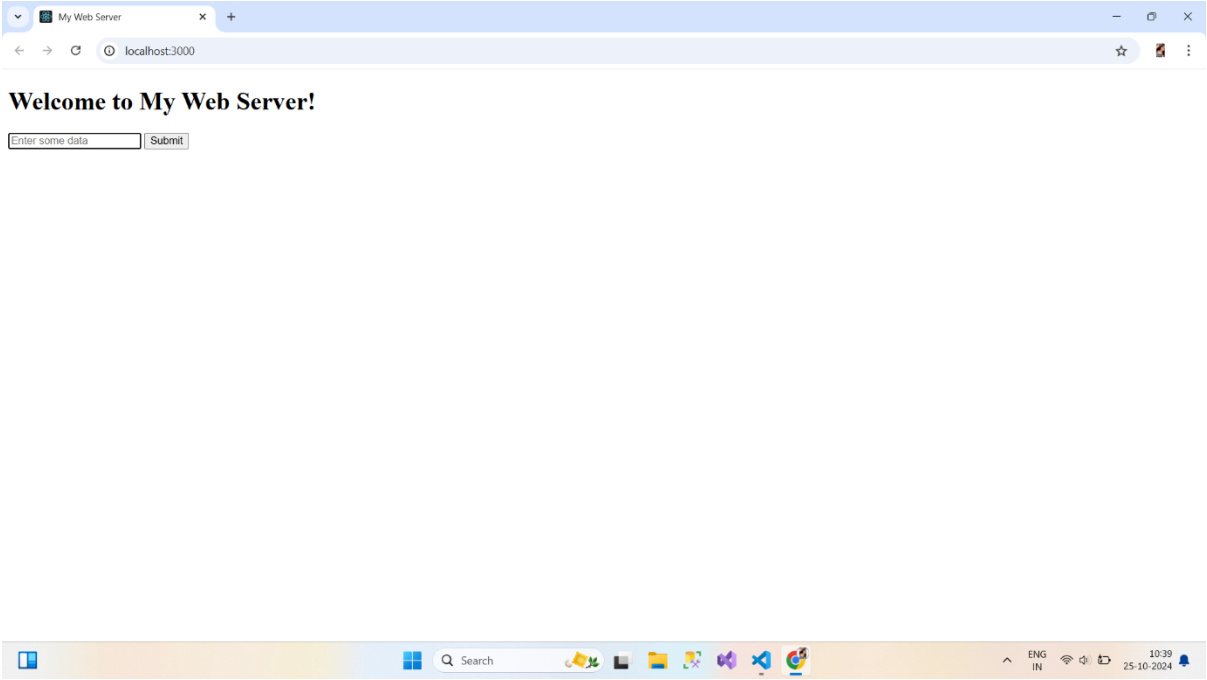
```
            '.otf': 'application/font-otf',
            '.txt': 'text/plain',
            '.xml': 'application/xml',
            '.pdf': 'application/pdf',
            '.zip': 'application/zip',
            '.css': 'text/css',
        };
        const contentType = mimeTypes[extname] || 'application/octet-stream';
        serveStaticFile(res, filePath, contentType);
    }
  }

  // Handle POST request
  else if (req.method === 'POST' && parsedUrl.pathname === '/submit') {
    let body = '';
    req.on('data', chunk => {
        body += chunk.toString(); // Convert Buffer to string
    });
    req.on('end', () => {
        const postData = querystring.parse(body);
        console.log('Received data:', postData.data);
        res.writeHead(200, { 'Content-Type': 'text/plain' });
        res.end('Data received: ' + postData.data);
    });
  } else {
    res.writeHead(404);
    res.end('404 Not Found');
  }
});
```

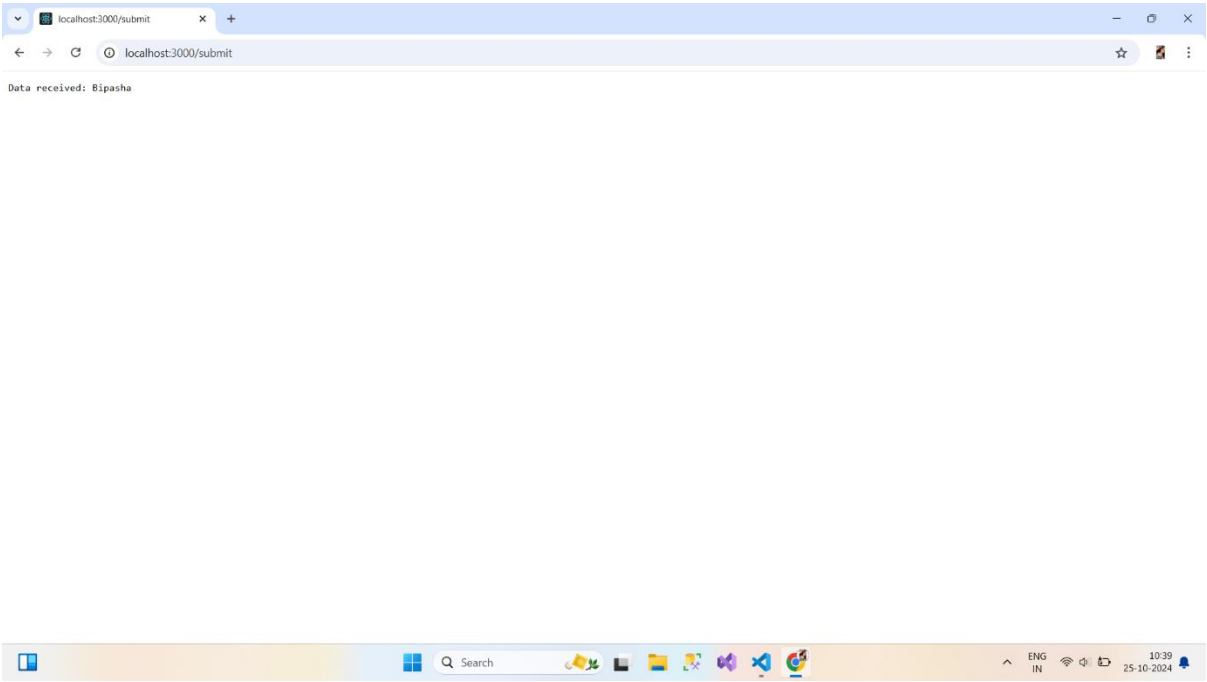
```
// Start the server
server.listen(PORT, () => {
  console.log(`Server is listening on http://localhost:${PORT}`);
});
```

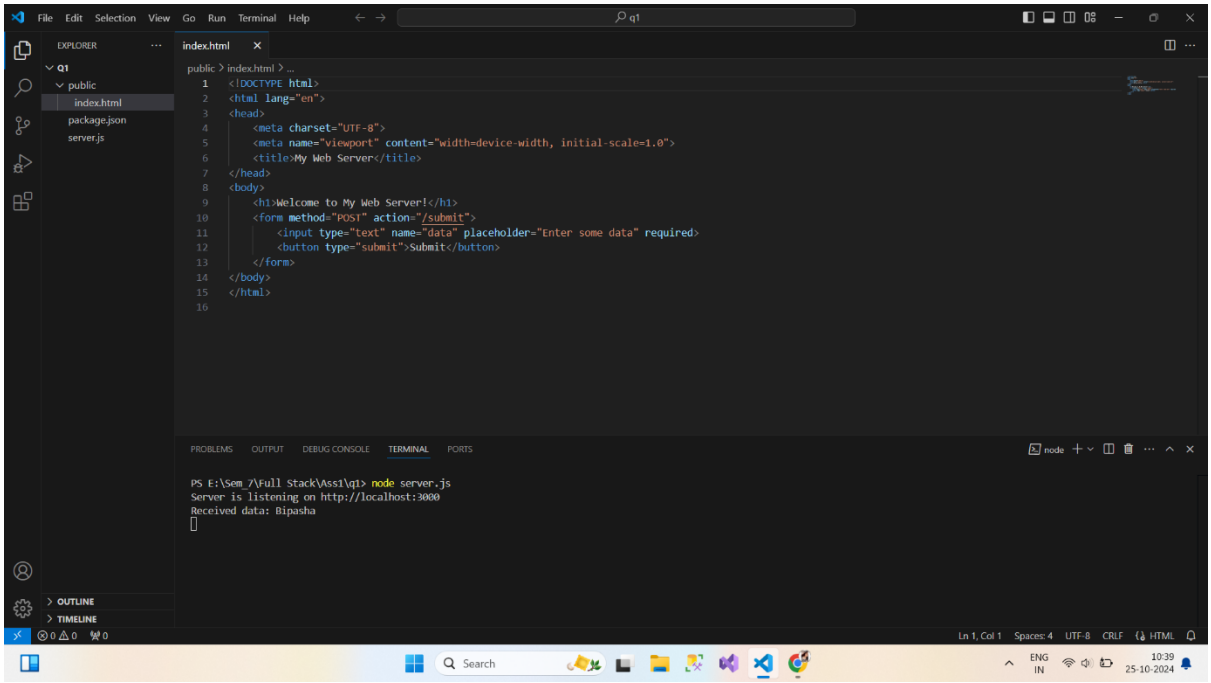
index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Web Server</title>
</head>
<body>
  <h1>Welcome to My Web Server!</h1>
  <form method="POST" action="/submit">
    <input type="text" name="data" placeholder="Enter some data" required>
    <button type="submit">Submit</button>
  </form>
</body>
</html>
```

Output :-


My Web Server
localhost:3000
Welcome to My Web Server!
Enter some data
Submit
Search
ENG
IN
10:39
25-10-2024


localhost:3000/submit
localhost:3000/submit
Data received: Bipasha
Search
ENG
IN
10:39
25-10-2024


File Edit Selection View Go Run Terminal Help
q1
EXPLORER
Q1
public
index.html
package.json
server.js
index.html
public > index.html > ...
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>My Web Server</title>
</head>
<body>
<h1>Welcome to My Web Server!</h1>
<form method="POST" action="/submit">
<input type="text" name="data" placeholder="Enter some data" required>
<button type="submit">Submit</button>
</form>
</body>
</html>
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
node
PS E:\Sem_7\Full Stack\Ass1\q1> node server.js
Server is listening on http://localhost:3000
Received data: Bipasha
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF HTML
Search
ENG IN
10:39
25-10-2024

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.)

server.js

```
const express = require('express');
const path = require('path');

const app = express();
const PORT = 3000;

// Serve static files from the public directory
app.use(express.static('public'));

// Route for /gethello
app.get('/gethello', (req, res) => {
  res.send('Hello NodeJS!!');
});

// Serve the HTML page
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'public', 'index.html'));
});
```

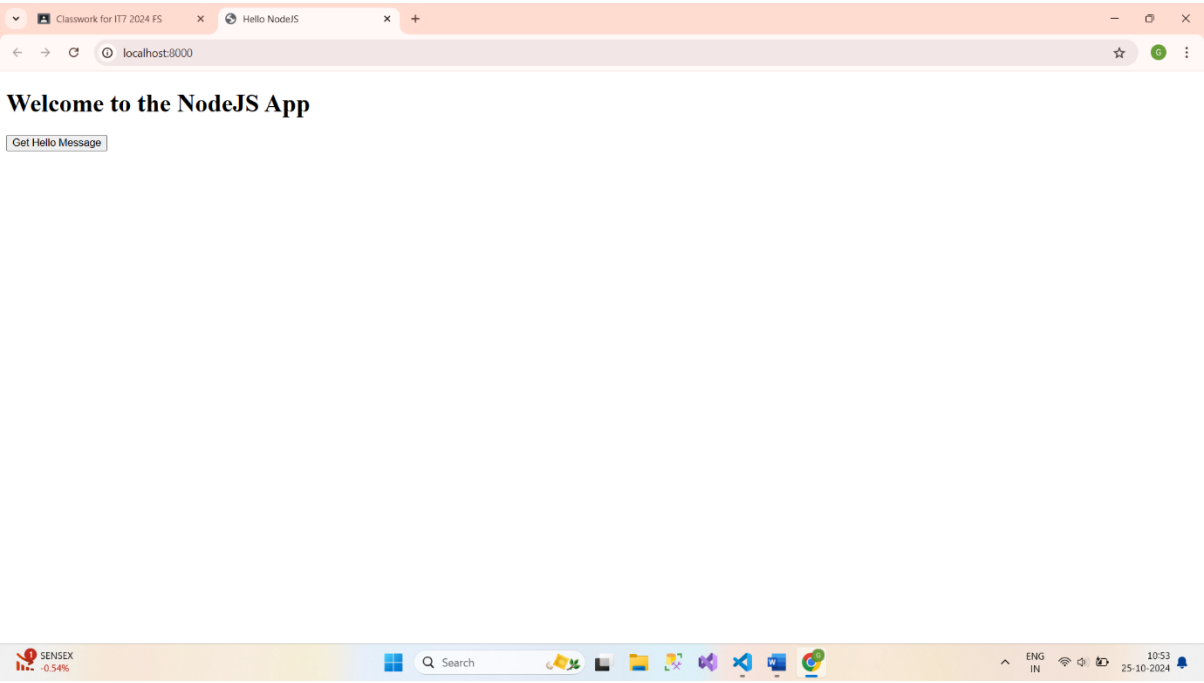
```
// Start the server
app.listen(PORT, () => {
  console.log(`Server is listening on http://localhost:${PORT}`);
});
```

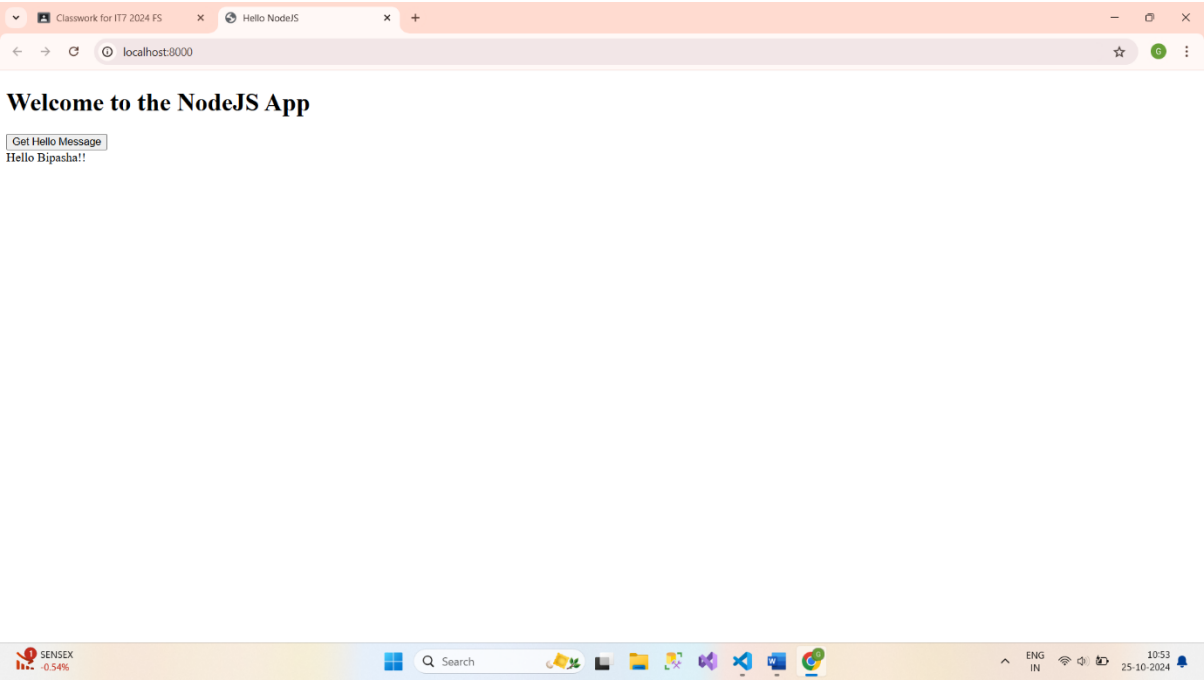
index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Hello NodeJS</title>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
  <h1>Welcome to the NodeJS App</h1>
  <button id="getHelloButton">Get Hello Message</button>
  <div id="response"></div>

  <script>
    $(document).ready(function() {
      $('#getHelloButton').click(function() {
        $.ajax({
          url: '/gethello',
          method: 'GET',
          success: function(data) {
            $('#response').text(data);
```

```
            },
            error: function() {
                $('#response').text('Error occurred while fetching data.');
            }
        });
    });
  });
  </script>
</body>
</html>
```

Output :-




Classwork for IT7 2024 FS
Hello NodeJS
localhost:8000
Welcome to the NodeJS App
Get Hello Message
Hello Bipasha!!
SENSEX
-0.54%
Search
ENG
IN
10:53
25-10-2024

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

app.js

```
const readline = require('readline');
const Chatbot = require('./chatbot');

// Initialize the chatbot with a specific domain
const chatbot = new Chatbot('Customer Support');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

console.log('Welcome to the chatbot application!');
console.log('Type "exit" to quit.\n');

const askQuestion = () => {
  rl.question('You: ', (input) => {
    if (input.toLowerCase() === 'exit') {
      console.log('Chatbot: Goodbye!');
      rl.close();
      return;
    }

    const response = chatbot.respond(input);
    console.log(`Chatbot: ${response}\n`);
```

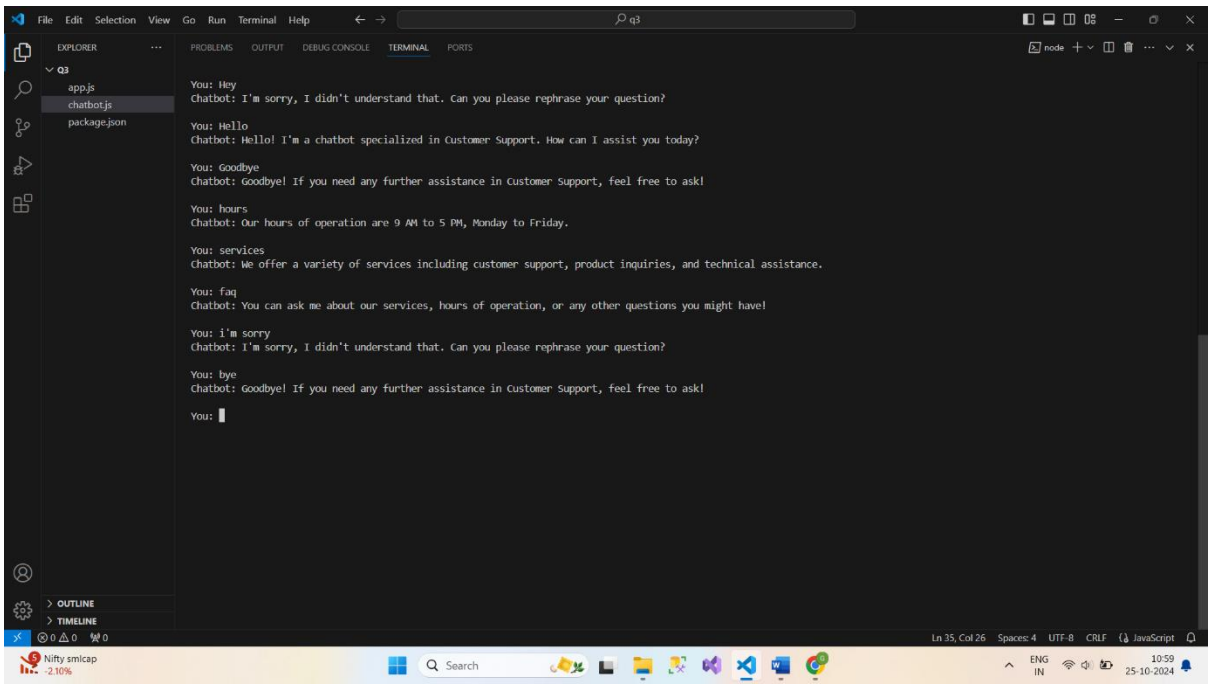
```
        askQuestion(); // Ask the next question
    });
};

// Start the conversation
askQuestion();
```

chatbot.js

```
// chatbot.js

class Chatbot {
    constructor(domain) {
        this.domain = domain;
        this.responses = {
            greeting: `Hello! I'm a chatbot specialized in ${this.domain}. How can I assist you today?`,
            farewell: `Goodbye! If you need any further assistance in ${this.domain}, feel free to ask!`,
            hours: `Our hours of operation are 9 AM to 5 PM, Monday to Friday.`,
            services: `We offer a variety of services including customer support, product inquiries, and technical assistance.`,
            faq: `You can ask me about our services, hours of operation, or any other questions you might have!`,
            default: `I'm sorry, I didn't understand that. Can you please rephrase your question?`
        };
    }

    respond(message) {
```

```
        const lowerMessage = message.toLowerCase();


        if (lowerMessage.includes('hello') || lowerMessage.includes('hi')) {
            return this.responses.greeting;
        } else if (lowerMessage.includes('bye') || lowerMessage.includes('goodbye')) {
            return this.responses.farewell;
        } else if (lowerMessage.includes('hours')) {
            return this.responses.hours;
        } else if (lowerMessage.includes('services') || lowerMessage.includes('what do you
offer')) {
            return this.responses.services;
        } else if (lowerMessage.includes('faq') || lowerMessage.includes('questions')) {
            return this.responses.faq;
        } else {
            return this.responses.default;
        }
    }
}


module.exports = Chatbot;
```

Output :-


File Edit Selection View Go Run Terminal Help
q3
EXPLORER
q3
app.js
chatbot.js
package.json
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
node
You: Hey
Chatbot: I'm sorry, I didn't understand that. Can you please rephrase your question?
You: Hello
Chatbot: Hello! I'm a chatbot specialized in Customer Support. How can I assist you today?
You: Goodbye
Chatbot: Goodbye! If you need any further assistance in Customer Support, feel free to ask!
You: hours
Chatbot: Our hours of operation are 9 AM to 5 PM, Monday to Friday.
You: services
Chatbot: We offer a variety of services including customer support, product inquiries, and technical assistance.
You: faq
Chatbot: You can ask me about our services, hours of operation, or any other questions you might have!
You: i'm sorry
Chatbot: I'm sorry, I didn't understand that. Can you please rephrase your question?
You: bye
Chatbot: Goodbye! If you need any further assistance in Customer Support, feel free to ask!
You:
OUTLINE
TIMELINE
Ln 35, Col 26 Spaces: 4 UTF-8 CRLF JavaScript
Nifty smlcap
-2.10%
Search
ENG
IN
10:59
25-10-2024

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

Server.js

```
// server.js
const express = require('express');
const path = require('path');
const WebSocket = require('ws');
const chatbot = require('./chatbot');

const app = express();
const PORT = 2000;

// Serve static HTML file
app.use(express.static(path.join(__dirname, 'public')));

// Create HTTP server and WebSocket server
const server = app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});

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

// WebSocket connection handler
wss.on('connection', (ws) => {
  console.log('New client connected!');

  ws.on('message', (message) => {
    console.log(`Received: ${message}`);
```

```
    const response = chatbot(message);
    ws.send(`Bot: ${response}`);
  });

  ws.on('close', () => {
    console.log('Client disconnected.');
  });
});
```

chatbot.js

```
// chatbot.js
const responses = {
    "hello": "Hi! How can I assist you?",
    "bye": "Goodbye!",
    "help": "I can assist with general queries."
  };

  function chatbot(input) {
    return responses[input.toLowerCase()] || "I didn't understand that.";
  }

  module.exports = chatbot;
```

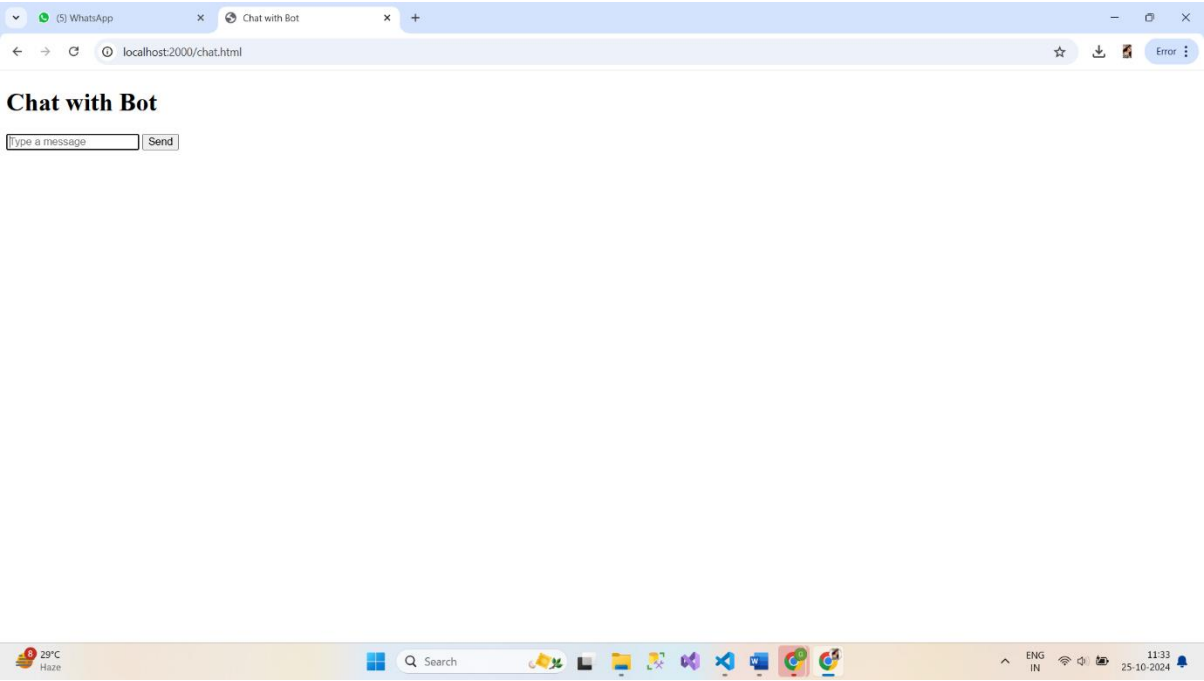
chat.html

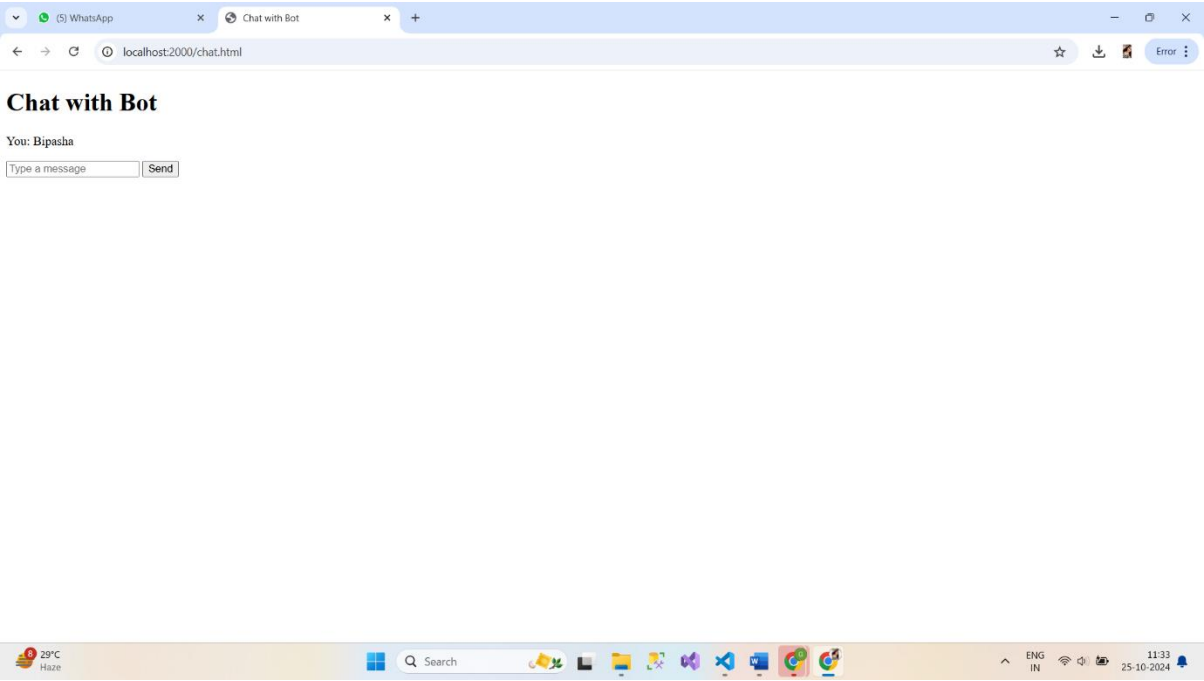
```
<!-- /public/index.html -->
<!DOCTYPE html>
```

```html
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Chat with Bot</title>
</head>
<body>
  <h1>Chat with Bot</h1>
  <div id="chatbox"></div>
  <input type="text" id="input" placeholder="Type a message" />
  <button onclick="sendMessage()">Send</button>

  <script>
    const ws = new WebSocket('ws://localhost:3000');

    ws.onmessage = function(event) {
      const chatbox = document.getElementById('chatbox');
      chatbox.innerHTML += `<p>${event.data}</p>`;
    };

    function sendMessage() {
      const input = document.getElementById('input');
      ws.send(input.value);
      const chatbox = document.getElementById('chatbox');
      chatbox.innerHTML += `<p>You: ${input.value}</p>`;
      input.value = ''; // Clear input field after sending
    }
  </script>
</body>
```

</html>

Output :-


(5) WhatsApp
Chat with Bot
localhost:2000/chat.html
Error
Chat with Bot
Type a message
Send
29°C
Haze
Search
ENG
IN
11:33
25-10-2024


(5) WhatsApp
Chat with Bot
localhost:2000/chat.html
Error
Chat with Bot
You: Bipasha
Type a message
Send
29°C
Haze
Search
ENG
IN
11:33
25-10-2024

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

file1.txt

Hello, Bipasha!

file2.txt

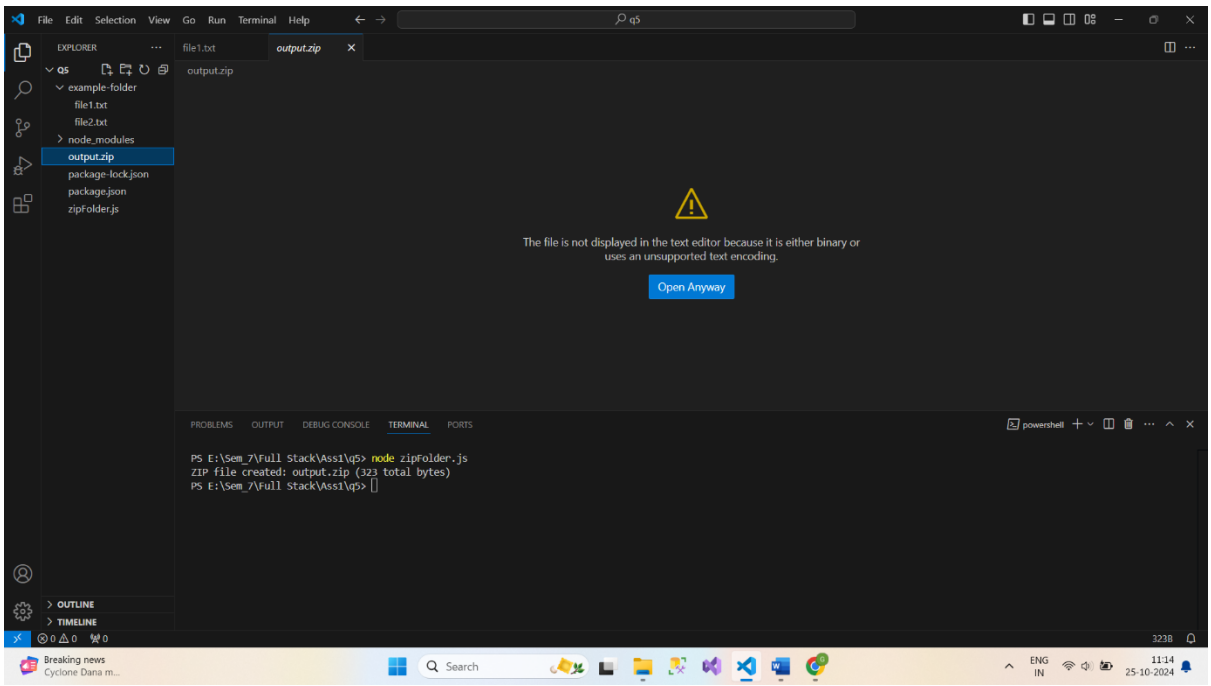
This is a test file.

zipFolder.js

```
const fs = require('fs-extra');
const archiver = require('archiver');

function zipFolder(sourceFolder, outPath) {
    const output = fs.createWriteStream(outPath);
    const archive = archiver('zip', {
        zlib: { level: 9 } // Set the compression level
    });

    output.on('close', () => {
        console.log(`ZIP file created: ${outPath} (${archive.pointer()} total bytes)`);
    });

    archive.on('error', (err) => {
        throw err;
    });

    archive.pipe(output);
    archive.directory(sourceFolder, false); // Include all files in the folder
```

```
    archive.finalize();
}

// Example usage
const folderToZip = 'example-folder'; // Change this to the folder you want to zip
const outputZipPath = 'output.zip'; // Name of the output zip file

zipFolder(folderToZip, outputZipPath);
```

output.zip



6. Write a program to extract a zip file.

file1.txt

Hello, Bipasha!

file2.txt

This is a test file.

extractZip.js

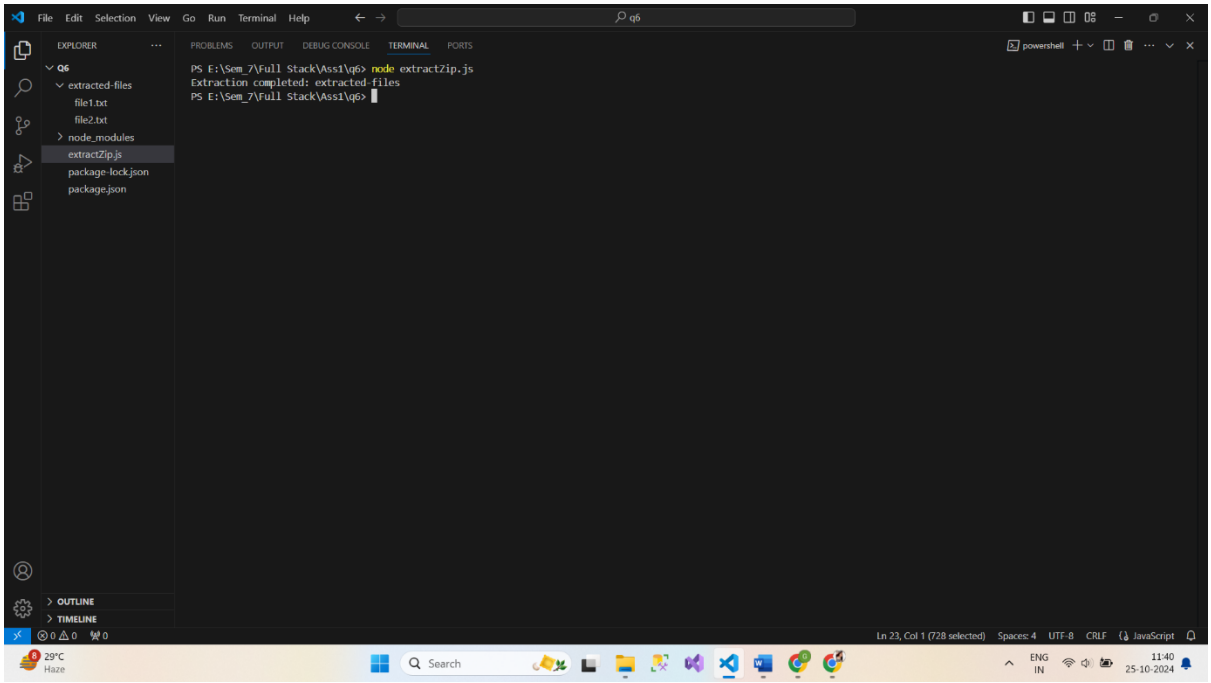
```
// extractZip.js

const fs = require('fs');
const unzipper = require('unzipper');

function extractZip(zipFilePath, outputFolder) {
  fs.createReadStream(zipFilePath)
    .pipe(unzipper.Extract({ path: outputFolder }))
    .on('close', () => {
      console.log(`Extraction completed: ${outputFolder}`);
    })
    .on('error', (err) => {
      console.error(`Error during extraction: ${err.message}`);
    });
}

// Example usage
const zipFilePath = '../q5/output.zip'; // Adjust the path if needed
```

; // Change this to the path of your zip file

const outputFolder = 'extracted-files'; // Folder where extracted files will be saved

extractZip(zipFilePath, outputFolder);

Output :-

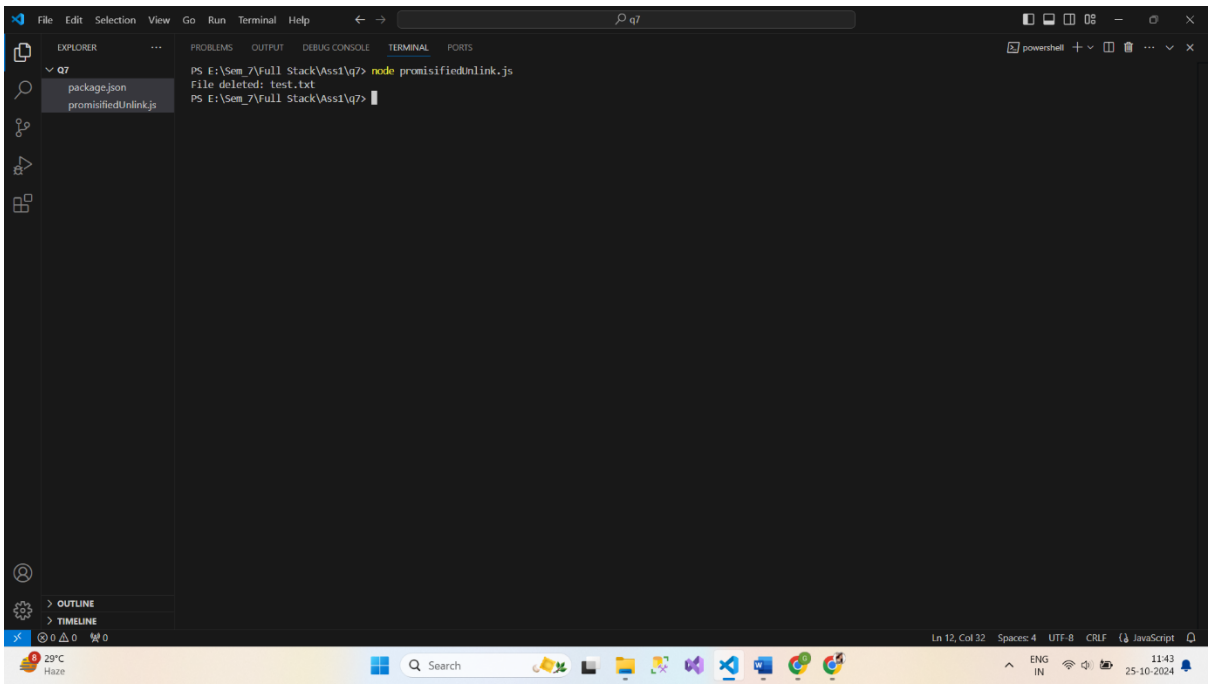


7. Write a program to promisify fs.unlink function and call it.

promisifiedUnlink.js

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

// Promisify the fs.unlink function
const unlink = util.promisify(fs.unlink);

// Function to delete a file
async function deleteFile(filePath) {
    try {
        await unlink(filePath);
        console.log(`File deleted: ${filePath}`);
    } catch (err) {
        console.error(`Error deleting file: ${err.message}`);
    }
}

// Example usage
const fileToDelete = 'test.txt'; // Change this to the file you want to delete

// Create a test file for demonstration
fs.writeFileSync(fileToDelete, 'This is a test file.');

deleteFile(fileToDelete);
```

Output :-


File Edit Selection View Go Run Terminal Help
q7
EXPLORER
Q7
package.json
promisifiedUnlink.js
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
PS E:\Sem_7\Full Stack\Ass1\q7> node promisifiedUnlink.js
File deleted: test.txt
PS E:\Sem_7\Full Stack\Ass1\q7>
OUTLINE
TIMELINE
Ln 12, Col 32 Spaces: 4 UTF-8 CRLF JavaScript
29°C
Haze
Search
ENG
IN
11:43
25-10-2024

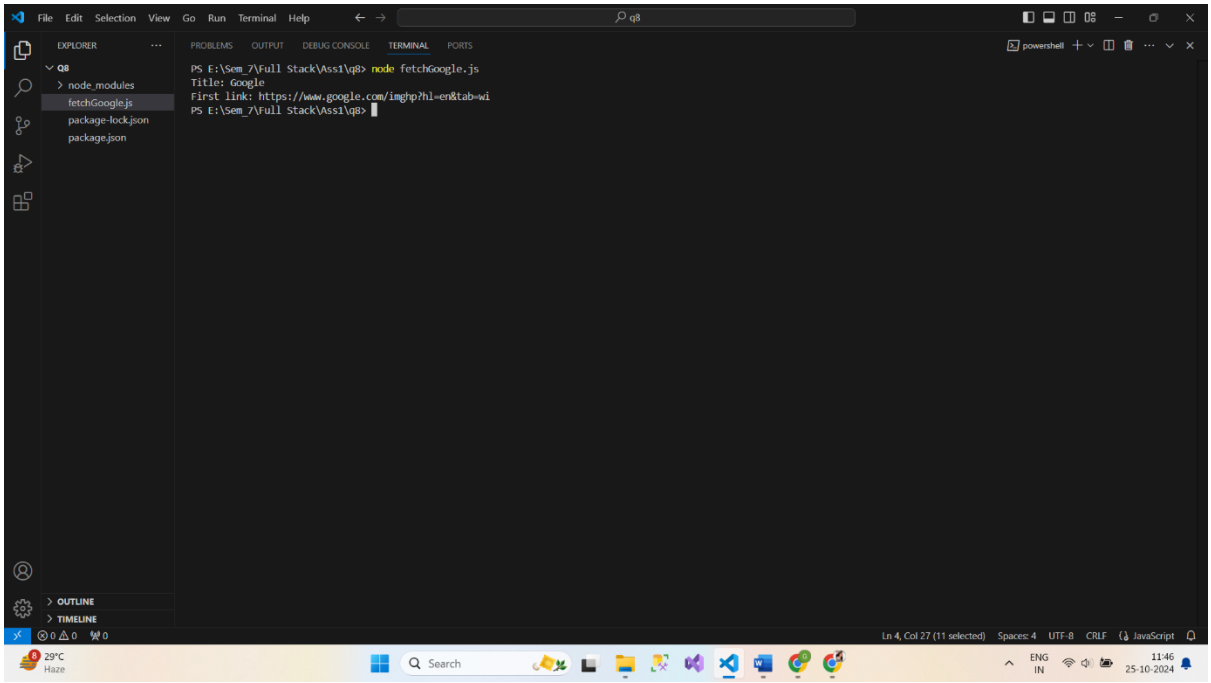
8. Fetch data of google page using note-fetch using async-await model.

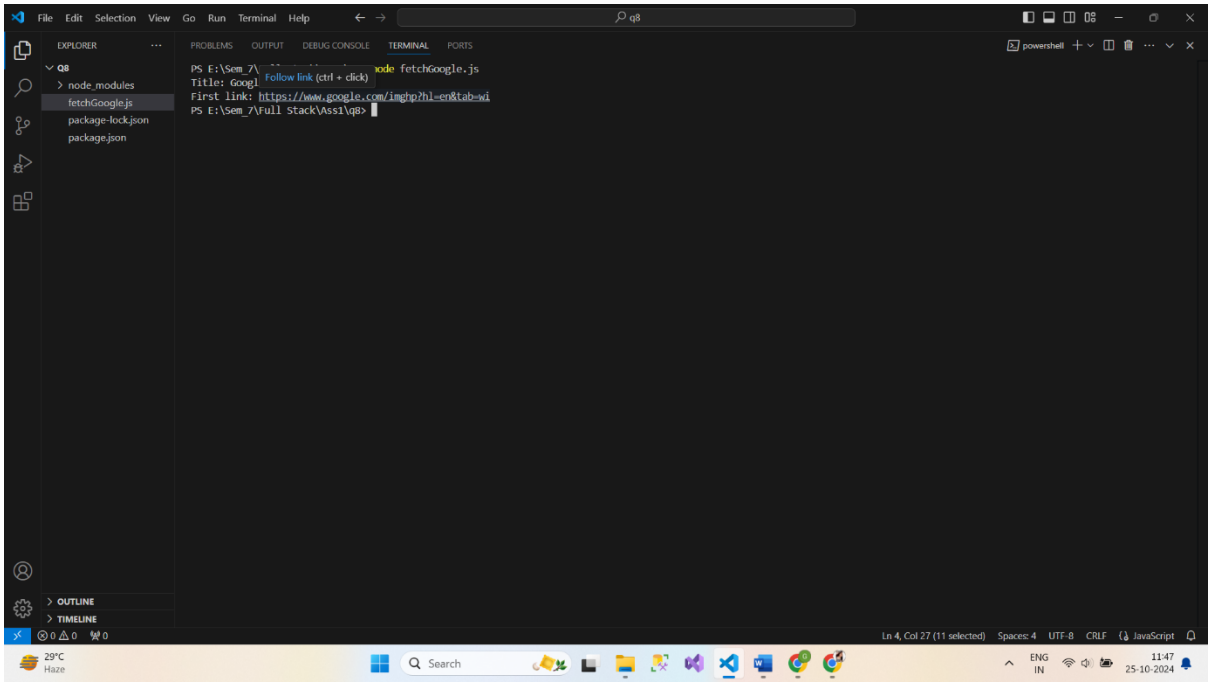
fetchGoogle.js

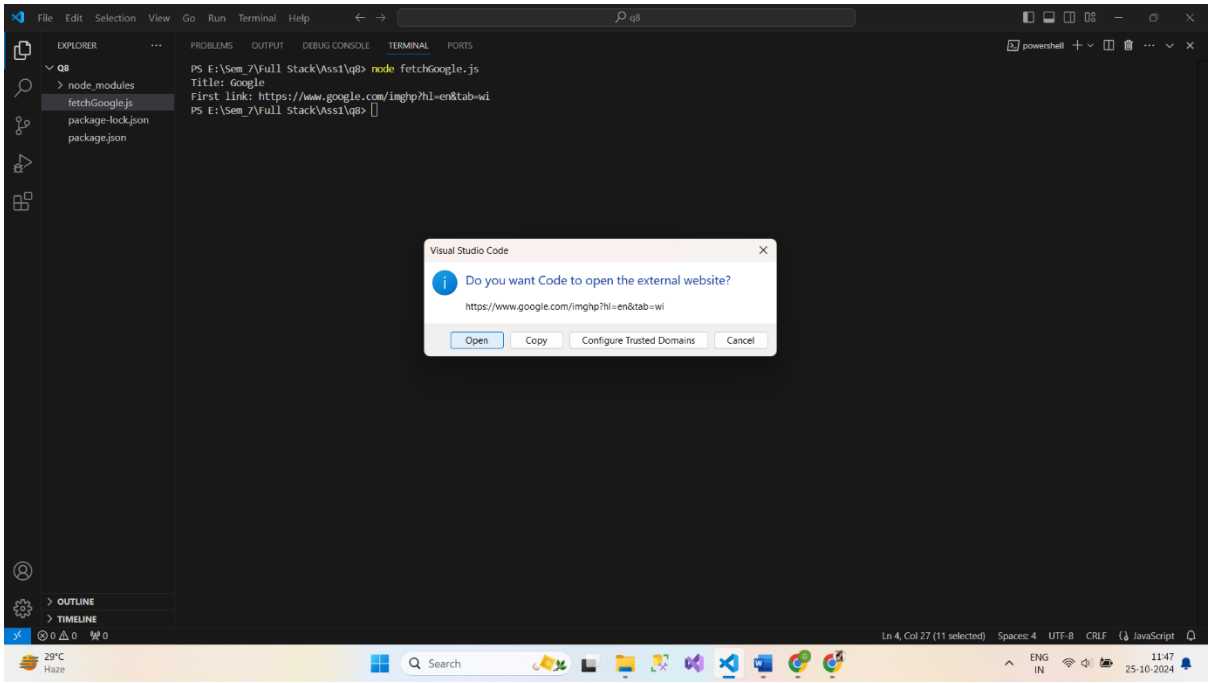
```
import fetch from 'node-fetch';
import * as cheerio from 'cheerio'; // Use named import

async function fetchGooglePage() {
  try {
    const response = await fetch('https://www.google.com');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const data = await response.text();
    const $ = cheerio.load(data);

    // Example: Get the title of the page
    const title = $('title').text();
    console.log(`Title: ${title}`);

    // Example: Get the first link
    const firstLink = $('a').first().attr('href');
    console.log(`First link: ${firstLink}`);
  } catch (error) {
    console.error(`Error fetching Google page: ${error.message}`);
  }
}
```

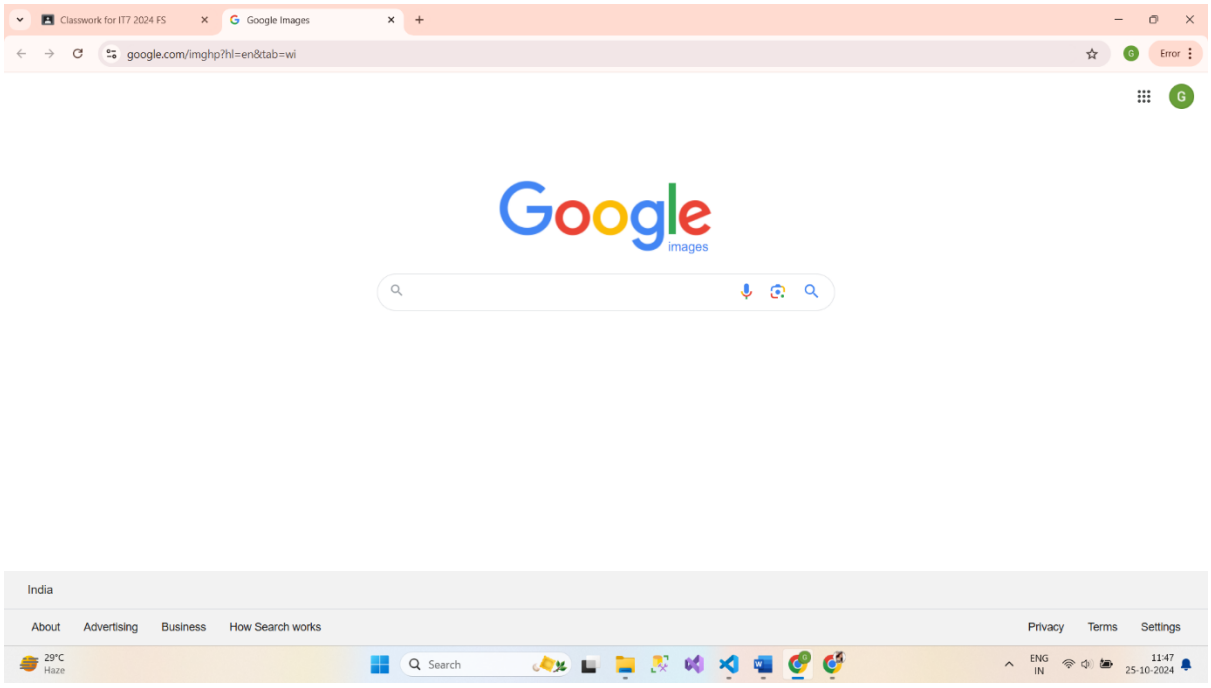
fetchGooglePage();

Output :-






File Edit Selection View Go Run Terminal Help
EXPLORER
Q8
node_modules
fetchGoogle.js
package-lock.json
package.json
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\Sem_7\Full Stack\Ass1\q8> node fetchGoogle.js
Title: Google
First link: https://www.google.com/imghp?hl=en&tab=wi
PS E:\Sem_7\Full Stack\Ass1\q8>
Visual Studio Code
Do you want Code to open the external website?
https://www.google.com/imghp?hl=en&tab=wi
Open
Copy
Configure Trusted Domains
Cancel
OUTLINE
TIMELINE
Ln 4, Col 27 (11 selected) Spaces: 4 UTF-8 CRLF JavaScript


Classwork for IT7 2024 FS
Google Images
google.com/imghp?hl=en&tab=wi
Google
images
India
About
Advertising
Business
How Search works
Privacy
Terms
Settings

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.

app.js

```
const mysql = require('mysql2/promise');

const dbConfig = {
   host: 'localhost',
   user: 'yourUsername',
   password: 'yourPassword',
   database: 'company'
};

async function connectDB() {
   const connection = await mysql.createConnection(dbConfig);
   console.log('Connected to the database.');
   return connection;
}

async function insertEmployee(connection, name, position, salary) {
   const query = 'INSERT INTO employee (name, position, salary) VALUES (?, ?, ?)';
   await connection.execute(query, [name, position, salary]);
   console.log('Employee record inserted.');
}

async function displayEmployees(connection) {
   const [rows] = await connection.execute('SELECT * FROM employee');
```

```
    console.log('Employee Records:');
    console.table(rows);
}

async function main() {
    const connection = await connectDB();

    try {

        await insertEmployee(connection, 'Bipasha Gosai', 'Developer', 60000);



        await displayEmployees(connection);
    } catch (error) {
        console.error('Error:', error);
    } finally {
        await connection.end();
        console.log('Connection closed.');
    }
}

main();
```

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

application.

script1.js

```
console.log('This is user-defined script 1.');
```

script2.js

```
console.log('I am Gosai Bipasha.');
```

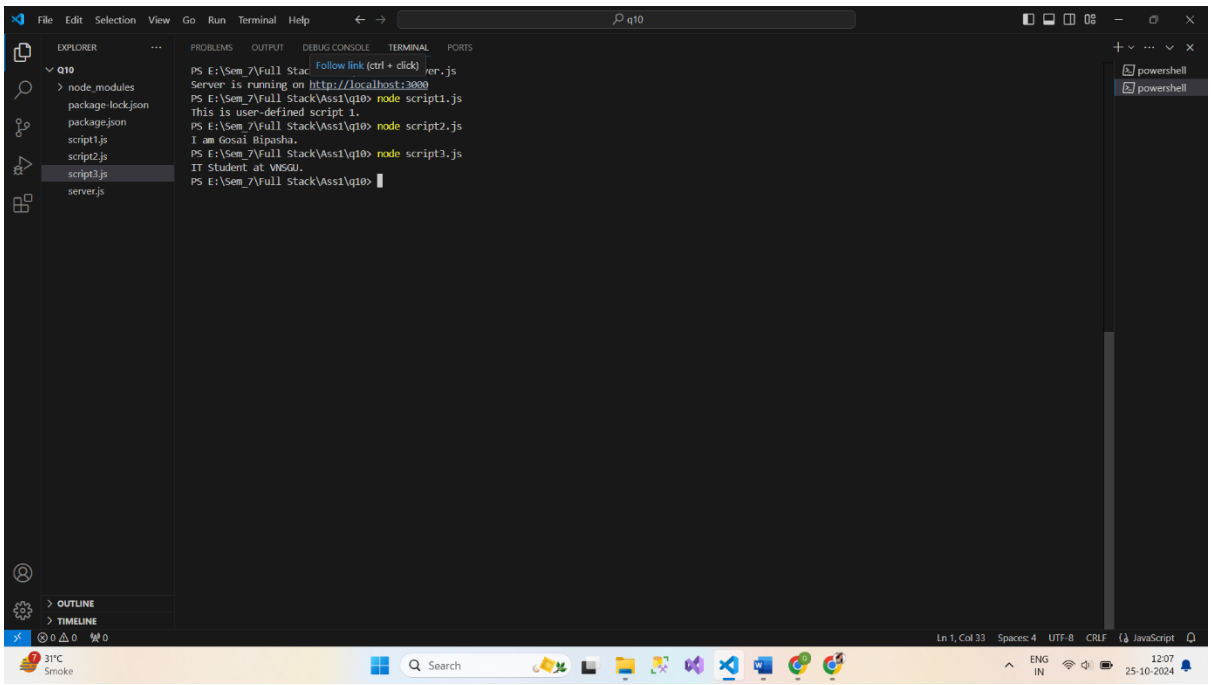
script3.js

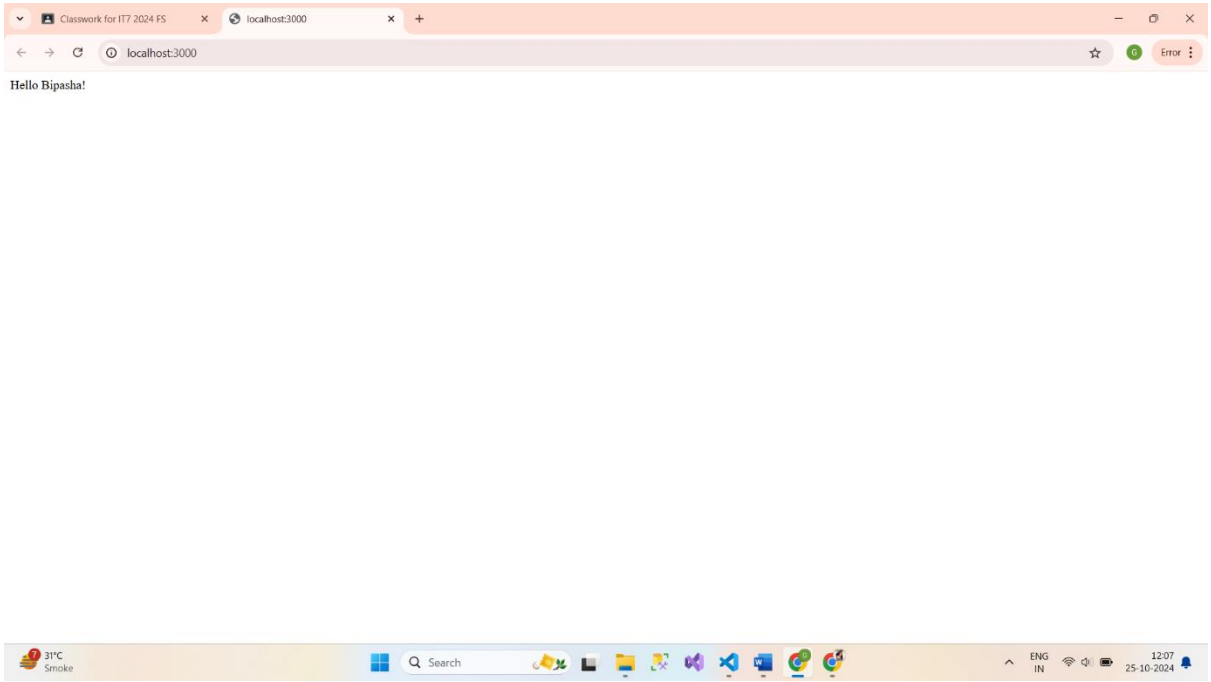
```
console.log('IT Student at VNSGU.');
```

server.js

```
import express from 'express';

const app = express();
const PORT = 3000;

app.get('/', (req, res) => {
   res.send('Hello Bipasha!');
});

app.listen(PORT, () => {
   console.log(`Server is running on http://localhost:${PORT}`);
});
```

Output :-


File Edit Selection View Go Run Terminal Help
q10
EXPLORER
Q10
node_modules
package-lock.json
package.json
script1.js
script2.js
script3.js
server.js
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Follow link (ctrl + click)
PS E:\Sem_7\Full Stac /er.js
Server is running on http://localhost:3000
PS E:\Sem_7\Full Stack\Ass1\q10> node script1.js
This is user-defined script 1.
PS E:\Sem_7\Full Stack\Ass1\q10> node script2.js
I am Gosai Bipasha.
PS E:\Sem_7\Full Stack\Ass1\q10> node script3.js
IT Student at VNSGU.
PS E:\Sem_7\Full Stack\Ass1\q10>
powershell
powershell
OUTLINE
TIMELINE
Ln 1, Col 33 Spaces: 4 UTF-8 CRLF JavaScript
31°C Smoke
Search
ENG IN
12:07 25-10-2024


Classwork for IT7 2024 FS
localhost:3000
localhost:3000
Error
Hello Bipasha!
31°C Smoke
Search
ENG IN
12:07 25-10-2024

11. Develop an application to show live cricket score.

server.js

```
// server.js
const express = require('express');
const app = express();
const PORT = process.env.PORT || 8000;

// Set EJS as the templating engine
app.set('view engine', 'ejs');

// Serve static files
app.use(express.static('public'));

// Sample static cricket scores
const scores = [
  {
    series: { name: 'IPL 2023' },
    team1: { name: 'Team A' },
    team2: { name: 'Team B' },
    status: 'Team A: 100/5 (18.0 overs) - Team B: 155/2 (17.0 overs) - Team B won by 8
wickets'
  },
  {
    series: { name: 'ODI Series' },
    team1: { name: 'Team C' },
    team2: { name: 'Team D' },
```

```
    status: 'Team C: 198/10 (40.0 overs) - Team D: 201/3 (35.0 overs) - Team D won by 7 wickets'
  }
];

// Home route
app.get('/', (req, res) => {
  res.render('index');
});

// Scores route
app.get('/scores', (req, res) => {
  res.render('scores', { scores });
});

// Start the server
app.listen(PORT, () => {
  console.log(`Server is running on http://localhost:${PORT}`);
});
```

index.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Live Cricket Score</title>
</head>
```

```
<body>
    <h1>Welcome to Live Cricket Score</h1>
    <a href="/scores">View Live Scores</a>
</body>
</html>
```

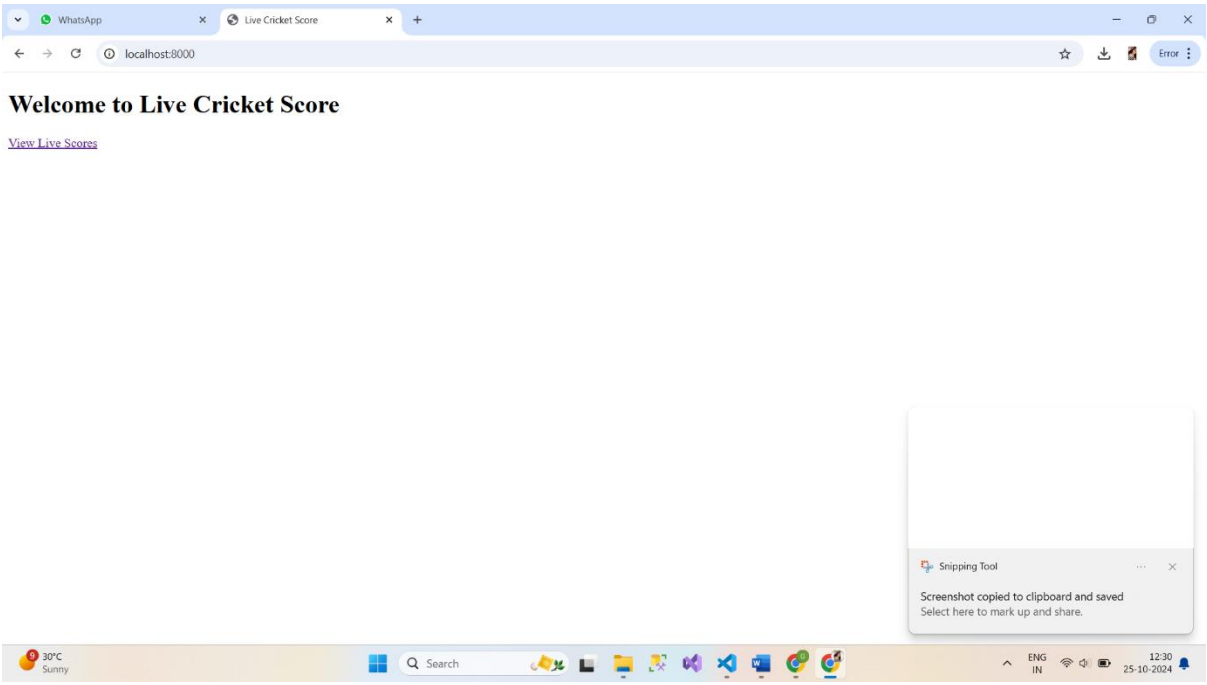
scores.js

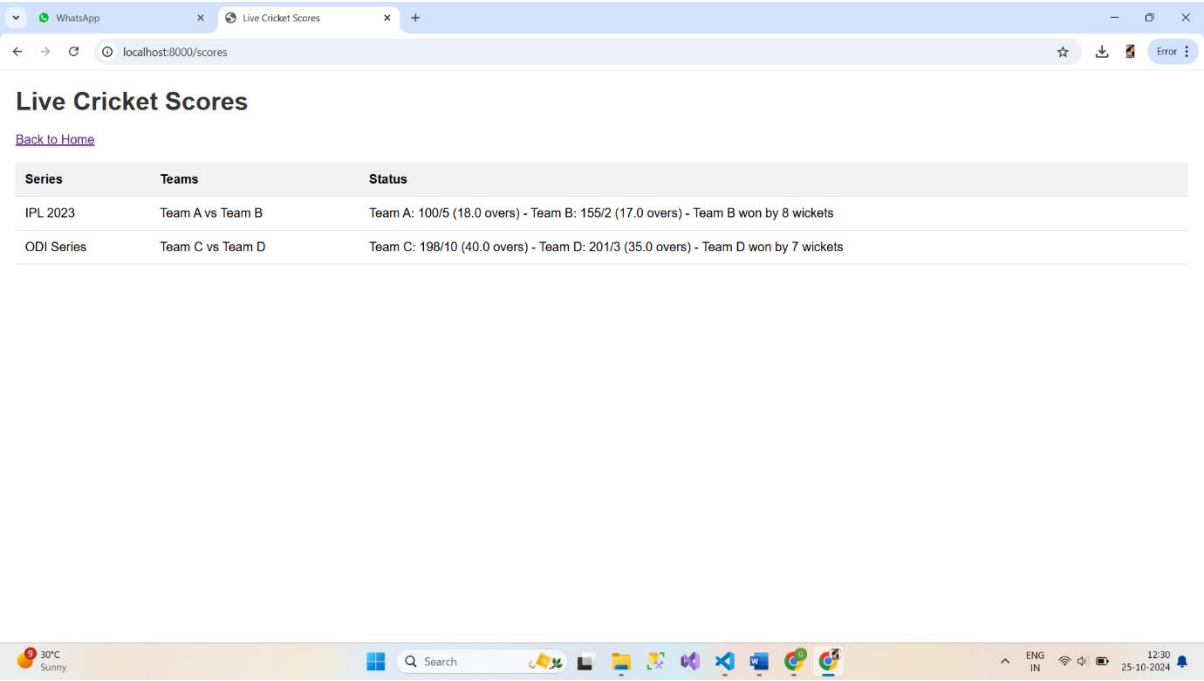
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Live Cricket Scores</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        table {
            width: 100%;
            border-collapse: collapse;
            margin-top: 20px;
        }
        th, td {
            padding: 12px;
            text-align: left;
            border-bottom: 1px solid #ddd;
        }
```

```
        th {
            background-color: #f2f2f2;
        }
        tr:hover {
            background-color: #f5f5f5;
        }
        h1 {
            color: #333;
        }
    </style>
</head>
<body>
    <h1>Live Cricket Scores</h1>
    <a href="/">Back to Home</a>
    <table>
        <thead>
            <tr>
                <th>Series</th>
                <th>Teams</th>
                <th>Status</th>
            </tr>
        </thead>
        <tbody>
            <% if (scores.length > 0) { %>
                <% scores.forEach(match => { %>
                    <tr>
                        <td><%= match.series.name %></td>
                        <td><%= match.team1.name %> vs <%= match.team2.name %></td>
                        <td><%= match.status %></td>
```

```
            </tr>
          <% }) %>
        <% } else { %>
          <tr>
            <td colspan="3">No live matches at the moment.</td>
          </tr>
        <% } %>
      </tbody>
    </table>
</body>
</html>
```

Output:-




WhatsApp
Live Cricket Scores
localhost:8000/scores
Error
Live Cricket Scores
Back to Home
Series
Teams
Status
IPL 2023
Team A vs Team B
Team A: 100/5 (18.0 overs) - Team B: 155/2 (17.0 overs) - Team B won by 8 wickets
ODI Series
Team C vs Team D
Team C: 198/10 (40.0 overs) - Team D: 201/3 (35.0 overs) - Team D won by 7 wickets
30°C
Sunny
Search
ENG
IN
12:30
25-10-2024