PRACTICAL ASSIGNMENT -1

Name : kanbi himanshi a.
Roll no : 021
Div : A
Subject : Full stack web development using node.js
Semester : 7th(M.sc.IT)
Github link:
https://github.com/himanshikanbi/Assignment1.git

Q:1. Develop a web server with following functionalities:
- Serve static resources. - Handle GET request.
- Handle POST request.

CODE:
Server.js

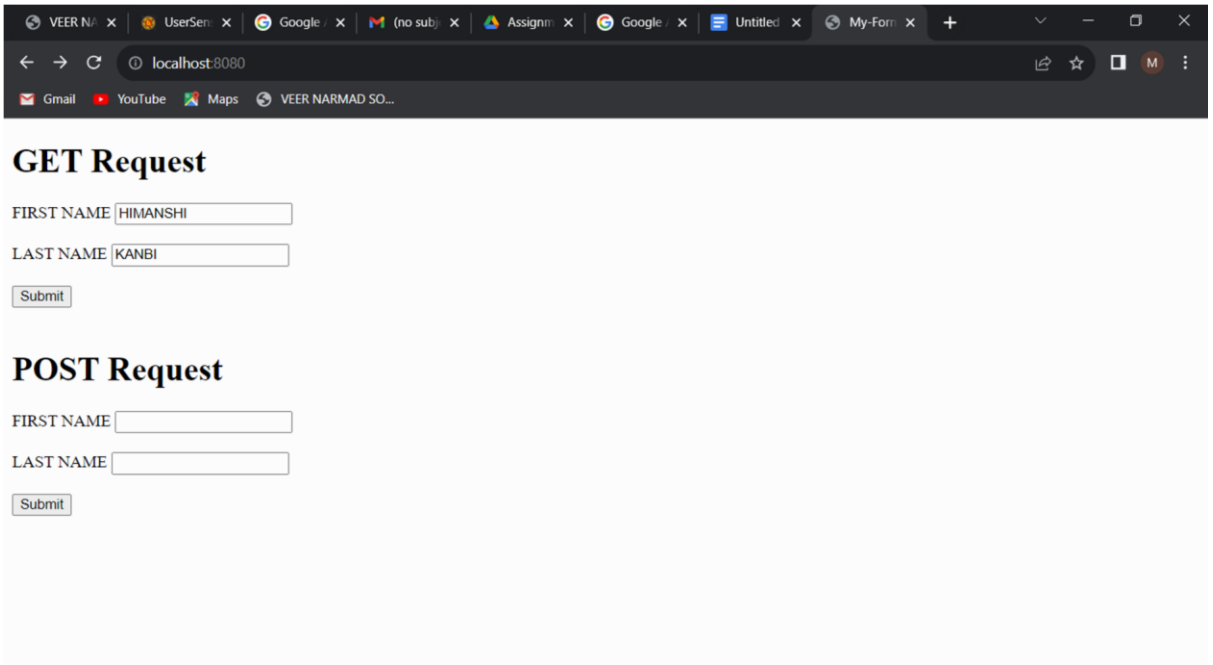
```
const http = require('http');
const fs = require('fs');
const url = require('url');


const server = http.createServer((req, res) => {
    var ul = url.parse(req.url, true);
    if(req.url == '/')
    {
        var path = "./public/index.html";
        fs.readFile(path, (err, data) => {
            if(err)
            { res.writeHead(404, {'Content-Type': 'text/html'});

                return res.end("404 page not found...");
            }
            res.writeHead(200, {'Content-Type': 'text/html'});
            res.write(data);
            res.end();
        });
    }
    else if(ul.pathname == '/process' && req.method == 'GET')
    {
        res.write("FIRST NAME: " + ul.query.fname + " \nLAST NAME: " +
ul.query.lname);
        res.end();
    }
    else if(ul.pathname == '/process' && req.method == 'POST')
    {
        let body = '';
        req.on('data', chunk => {
            body += chunk.toString();
        });
        req.on('end', () => {
            res.end(body);
```

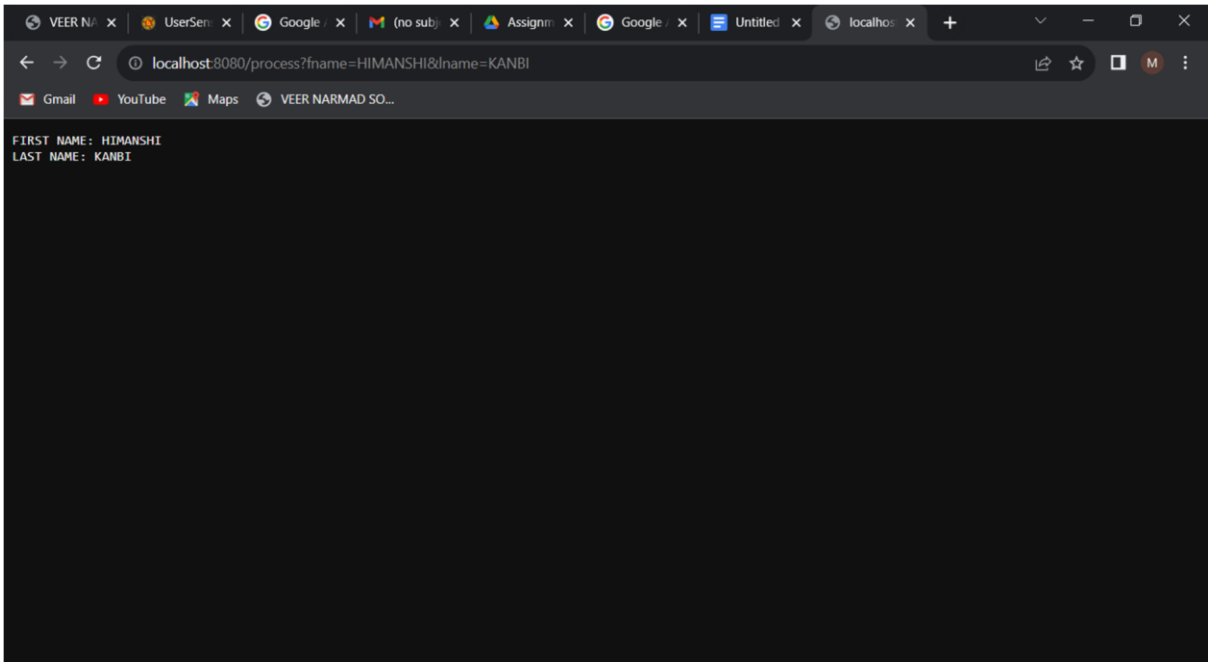
```
        });
    }
});
server.listen(8080);


console.log("The Server is Running on 8080");
```

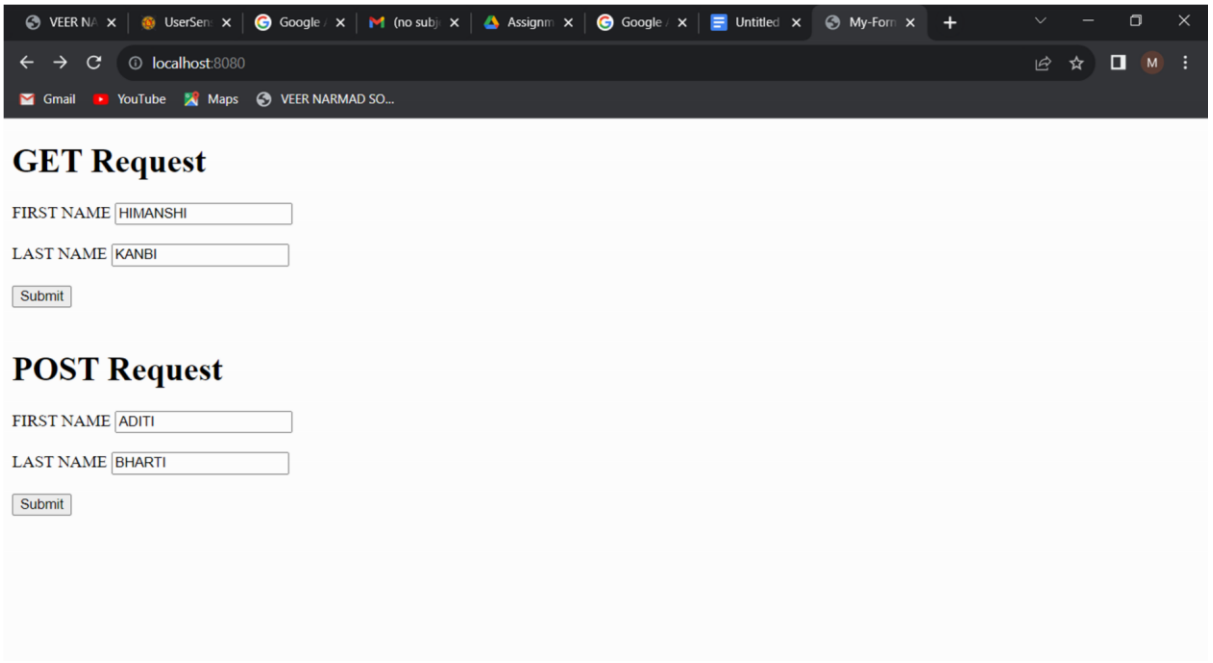
Index.html

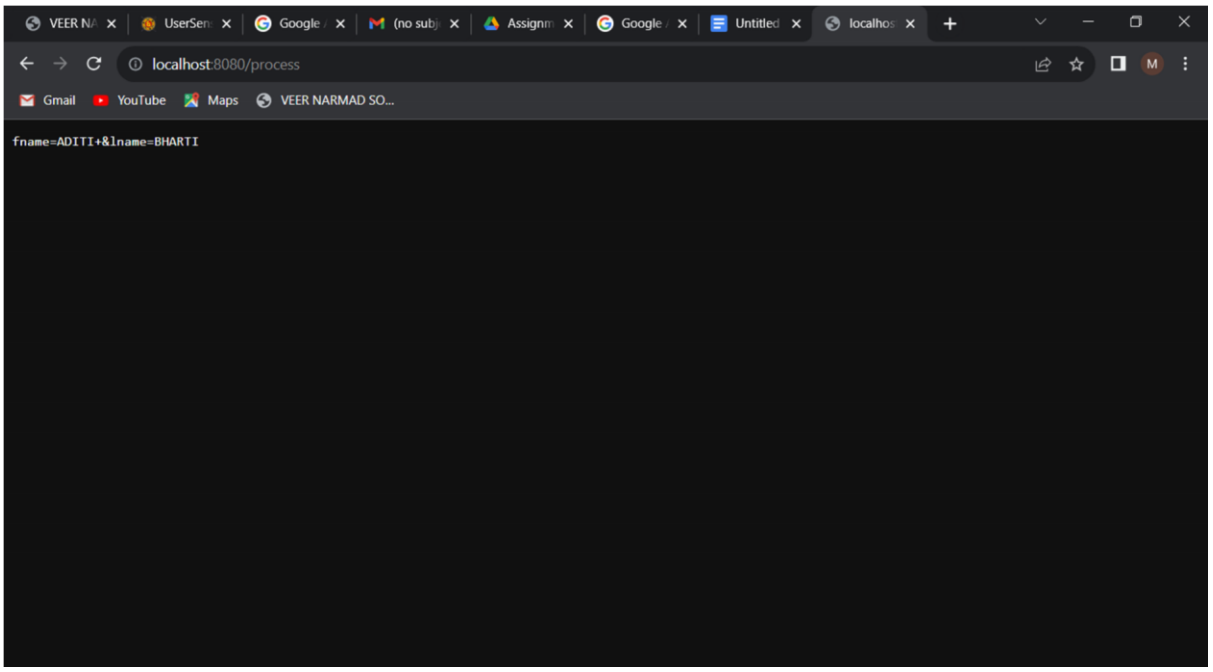
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>My-Form</title>
</head>
<body>
    <h1>GET Request</h1>
    <form action="/process" method="GET">
        FIRST NAME <input type="text" name="fname"/><br/><br/>
        LAST NAME <input type="text" name="lname"/><br/><br/>
        <input type="submit" value="Submit"/><br/><br/>
    </form>
    <h1>POST Request</h1>
    <form action="/process" method="POST">
        FIRST NAME <input type="text" name="fname"/><br/><br/>
        LAST NAME <input type="text" name="lname"/><br/><br/>
        <input type="submit" value="Submit"/>
    </form>
</body>
</html>
```

OUTPUT:


localhost:8080
Gmail
YouTube
Maps
VEER NARMAD SO...
GET Request
FIRST NAME HIMANSHI
LAST NAME KANBI
Submit
POST Request
FIRST NAME
LAST NAME
Submit


localhost:8080/process?fname=HIMANSHI&lname=KANBI
Gmail
YouTube
Maps
VEER NARMAD SO...
FIRST NAME: HIMANSHI
LAST NAME: KANBI





Q: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:

server.js

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

```
const fs=require('fs');
http.createServer((req, res) => {
    if (req.method === 'GET') {
        if (req.url === '/') {
            res.end("NODE JS");
        }
        if (req.url === '/gethello') {
            fs.readFile('./public/hello.html',(err,data)=>{
                if(err)
                {
                    return res.send("Something went wrong!!");
                }
                else{
                    res.writeHead(200,{
                        'Content-Type':'text/html'});
                    res.write(data);
                    return res.end();
                }
            })
        }
        if (req.url === '/ajaxcall') {
            fs.readFile('./public/ajaxcall.html',(err,data)=>{
                if(err)
                {
                    return res.send("Something went wrong!!");
                }
                else{
                    res.writeHead(200,{
                        'Content-Type':'text/html'});
                    res.write(data);
                    return res.end();
                }
            })
        }
    }


}).listen(8000, () => {
    console.log("server listening on port 8000");
```

```
})
```

Home.html

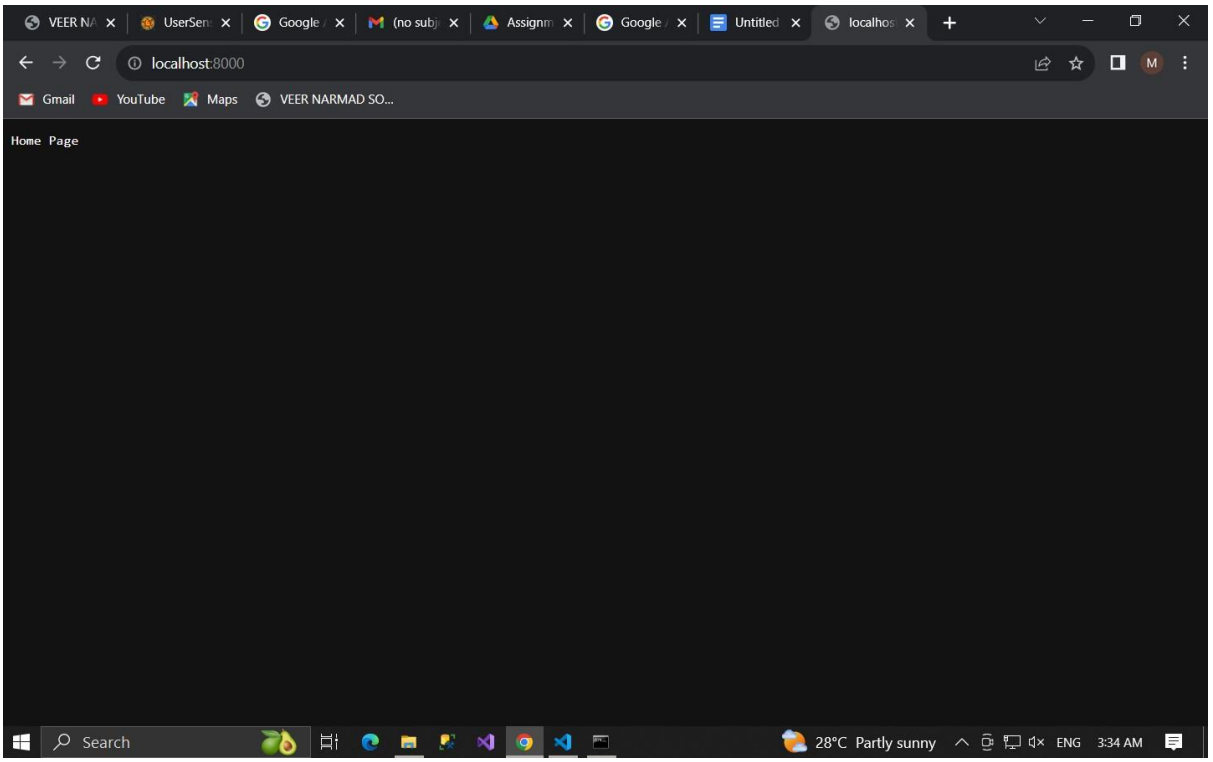
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h4>Hello NodeJS!!</h4>
</body>
</html>
```

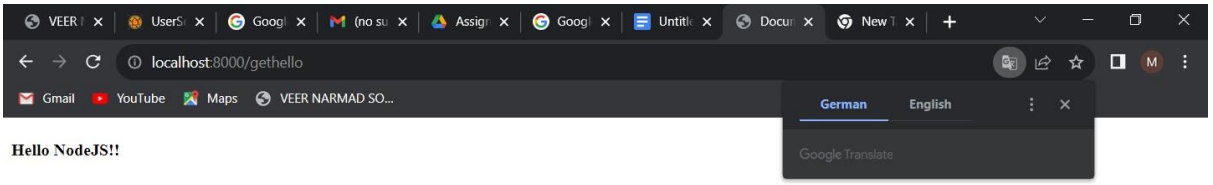
Ajax.html

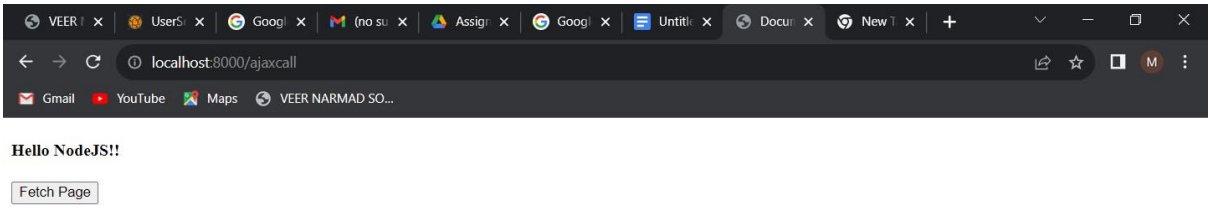
```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>


<body>
    <div id="page_content">


    </div>
    <button onclick="loadData()">Fetch Page</button>
    <script>
        function loadData() {
            var xhttp = new XMLHttpRequest();
            xhttp.onreadystatechange = function () {
                if (this.readyState == 4 && this.status == 200) {
```

```
                    document.getElementById("page_content").innerHTML =
this.responseText;
                }
            };
            xhttp.open("GET", "/gethello", true);
            xhttp.send();
        }
    </script>
</body>


</html>
```

OUTPUT:







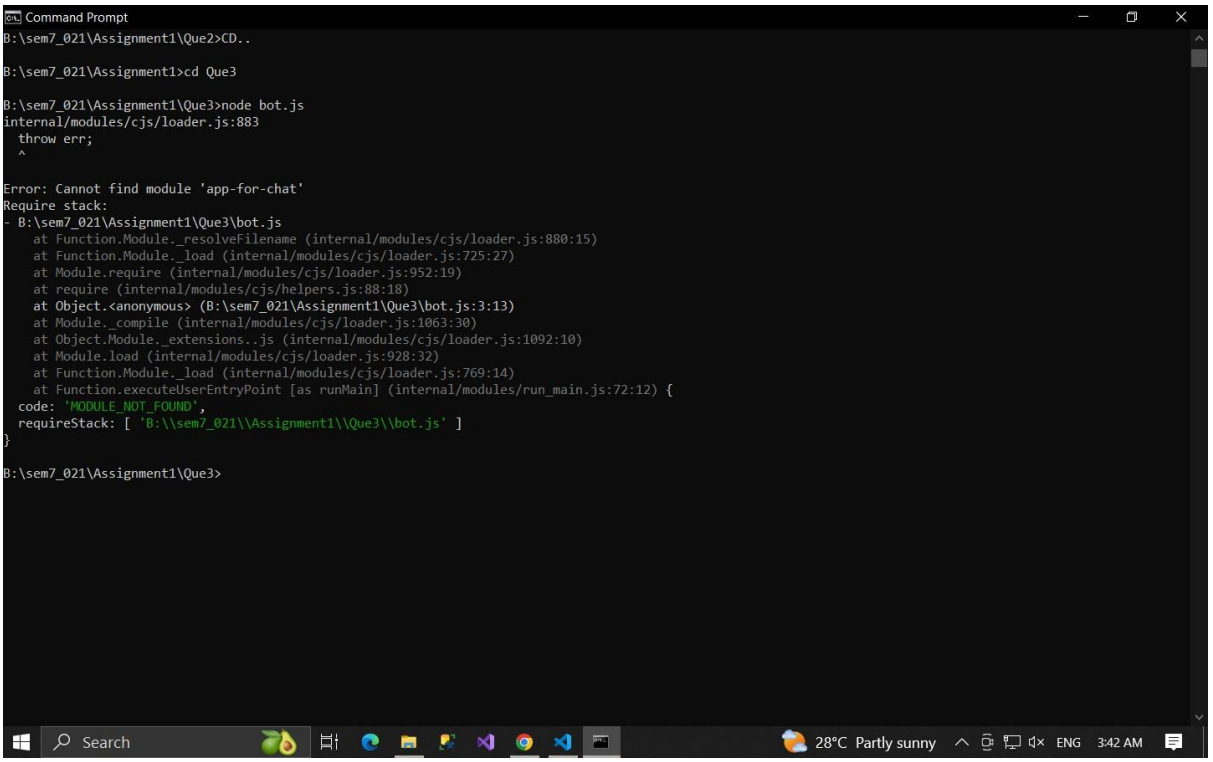
Q:3. Develop a module for domain specific chatbot and use it in a command line application.

CODE:

```
const readline = require('readline');
```

```
var rl = readline.createInterface({input: process.stdin, output:
process.stdout, terminal: false});
const app = require('app-for-chat');
```

OUTPUT:



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

CODE:

Server.js

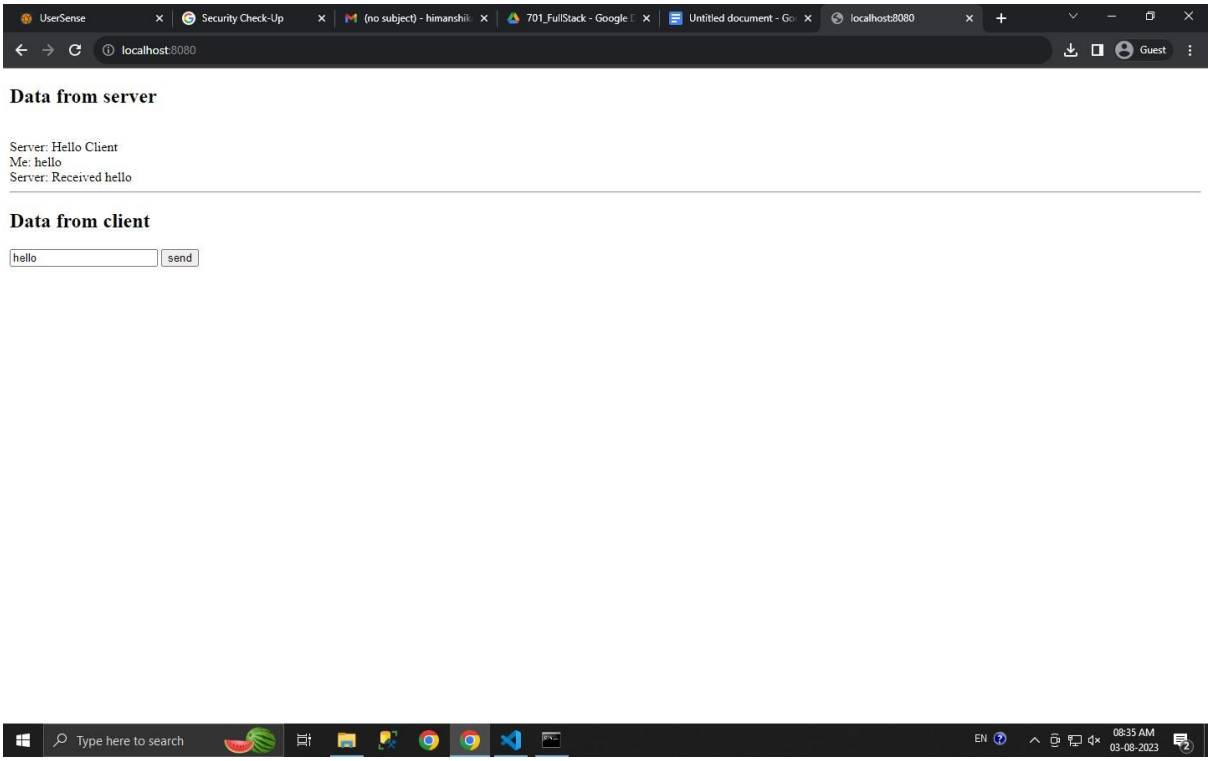
```
const WebSocket = require('ws')
var http = require('http');
var fs = require('fs');


var httpserver = http.createServer(function(request, response)
{
  if(request.url=="/")
  {
    fs.readFile("./public/index.html",(err,data)=>{
        response.write(data)
        response.end();
    })
  }
}).listen(8080, function() {
    console.log((new Date()) +
```

```
      ' Server is listening on port 8080');
});
const wss=new WebSocket.Server({server:httpserver})
wss.on("connection",(clientws)=>{
  clientws.send("Hello Client")
  clientws.on("message",(msg)=>{
    console.log("Received "+msg)
    clientws.send("Received "+ msg)
  })
})
```

Index.js

```
<!DOCTYPE html >
<html>
    <body>
<script language="javascript">
var ws = new WebSocket('ws://localhost:8080');
ws.addEventListener("message", function(msg1) {
  var msg = msg1.data;
  document.getElementById('chatlog').innerHTML+='<br>Server: '+ msg;
});


function sendMessage(){
    var message = document.getElementById('message').value;
    document.getElementById('chatlog').innerHTML+='<br> Me: '+
    message; ws.send(message);
}
</script>
<h2>Data from server</h2>
        <div id="chatlog"></div>
<hr/>
<h2>Data from client</h2>
    <input type="text" id="message" />
        <input type="button" id="b1" onclick="sendMessage()"
value="send" />
    </body>
</html>
```

OUTPUT:

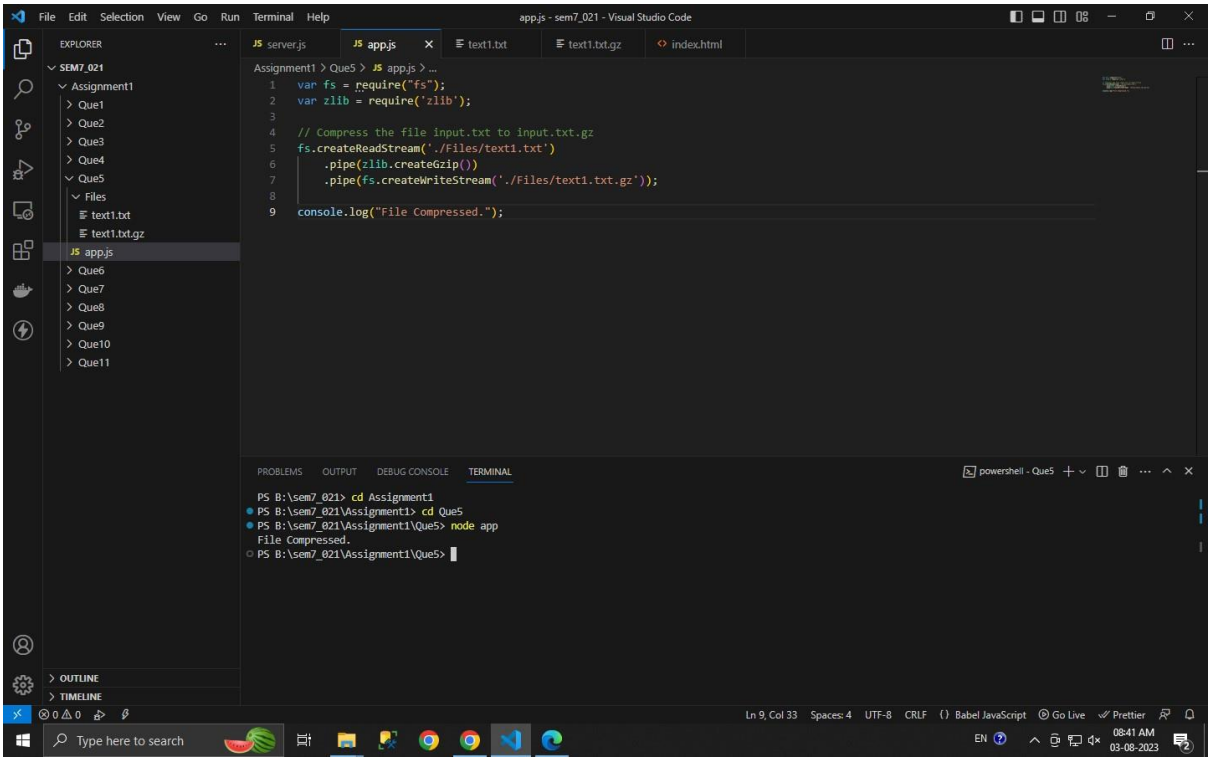


Q:5. Write a program to create a compressed zip file for a folder.

CODE:

```
var fs = require("fs");
var zlib = require('zlib');


// Compress the file input.txt to input.txt.gz
fs.createReadStream('./Files/text1.txt')
    .pipe(zlib.createGzip())
    .pipe(fs.createWriteStream('./Files/text1.txt.gz'));


console.log("File Compressed.");
```

OUTPUT:

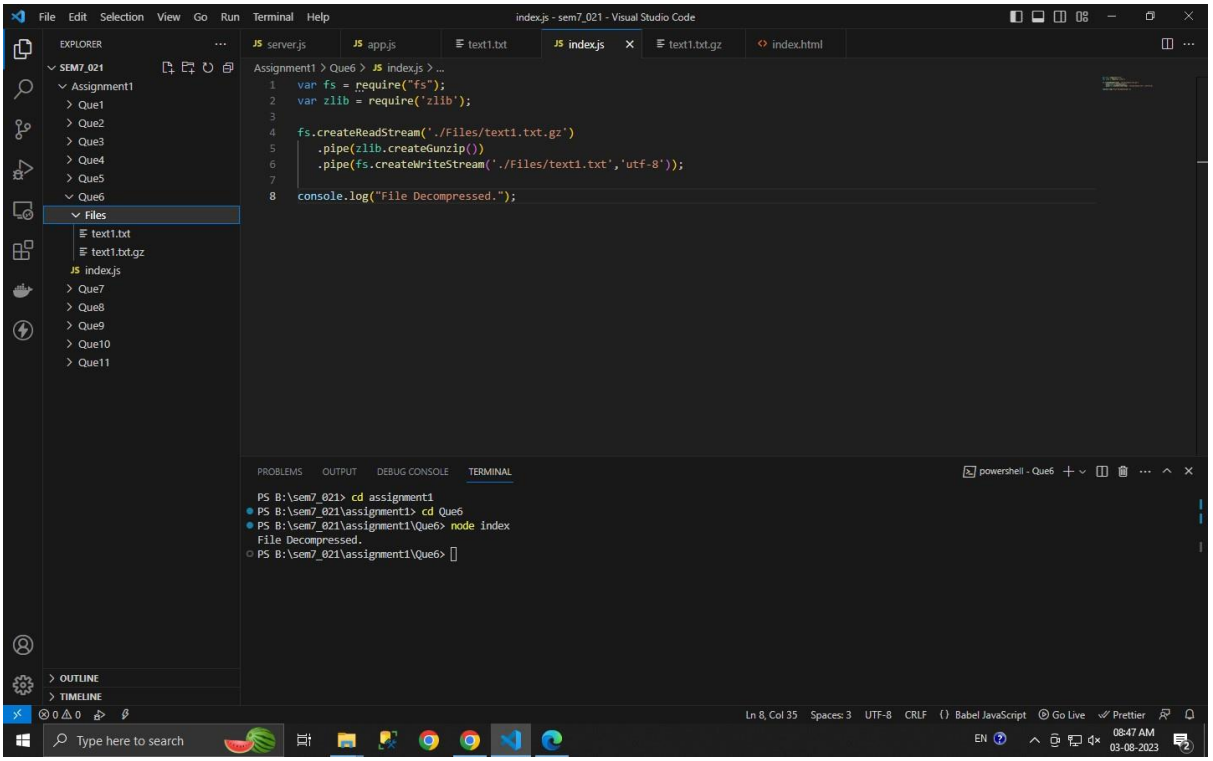


Q:6. Write a program to extract a zip file.

CODE:

```
var fs = require("fs");
var zlib = require('zlib');


fs.createReadStream('./Files/text1.txt.gz')
   .pipe(zlib.createGunzip())
   .pipe(fs.createWriteStream('./Files/text1.txt','utf-8'));


console.log("File Decompressed.");
```

OUTPUT:



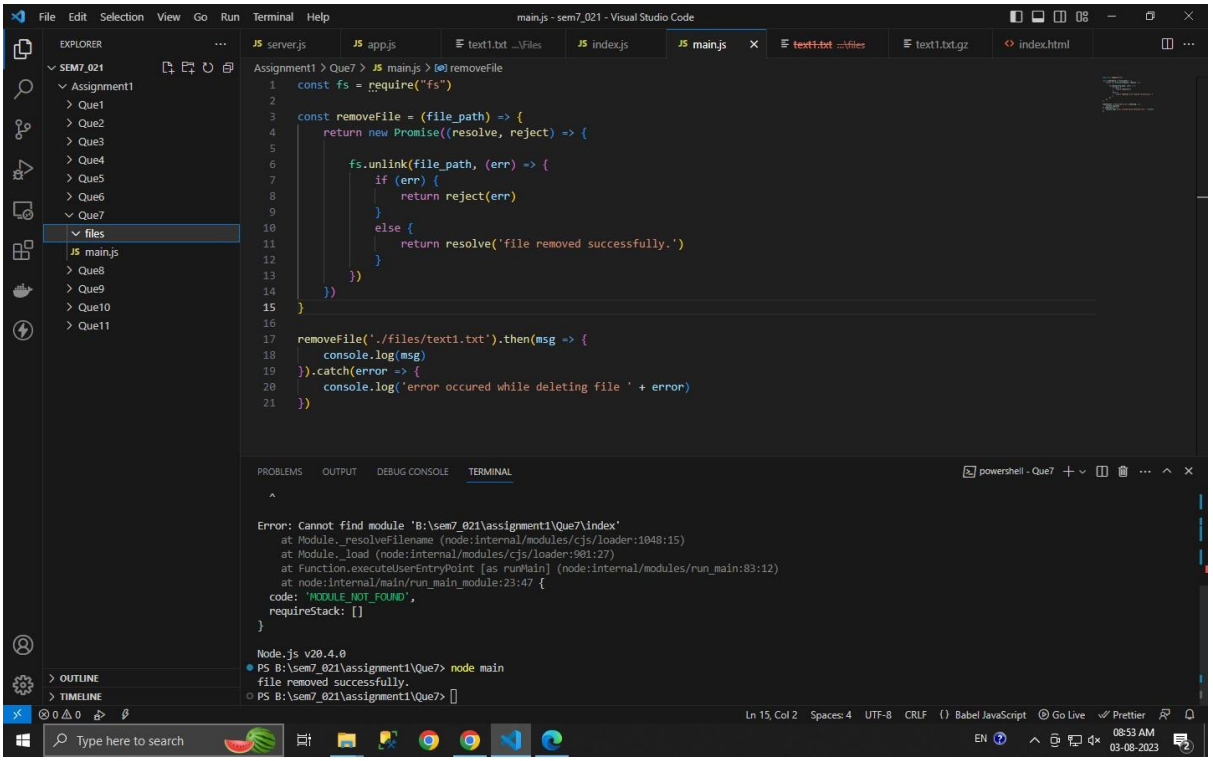
Q:7. Write a program to promisify fs.unlink function and call it.

CODE:

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


const removeFile = (file_path) => {
    return new Promise((resolve, reject) => {

        fs.unlink(file_path, (err) => {
            if (err) {
                return reject(err)
            }
            else {
                return resolve('file removed successfully.')
            }
        })
    })
}
```

```
removeFile('./files/text1.txt').then(msg => {

    console.log(msg)

}).catch(error => { console.log('error occured while
    deleting file ' + error)

})
```

OUTPUT:



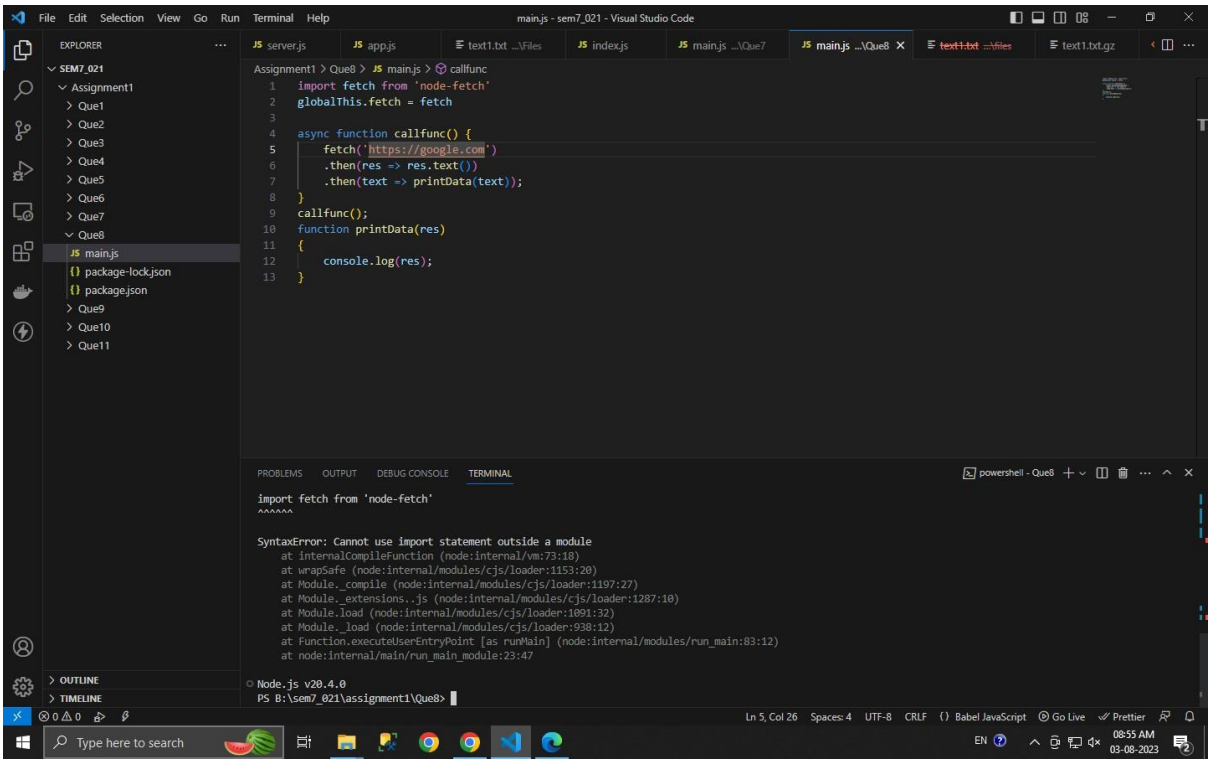
Q:8. Fetch data of google page using note-fetch using async-await model.

CODE:

```
import fetch from 'node-fetch'
globalThis.fetch = fetch



async function callfunc() {
    fetch('https://google.com')
    .then(res => res.text())
    .then(text => printData(text));
}
callfunc();

function printData(res)
{
    console.log(res);
```

```
}
```

OUTPUT:



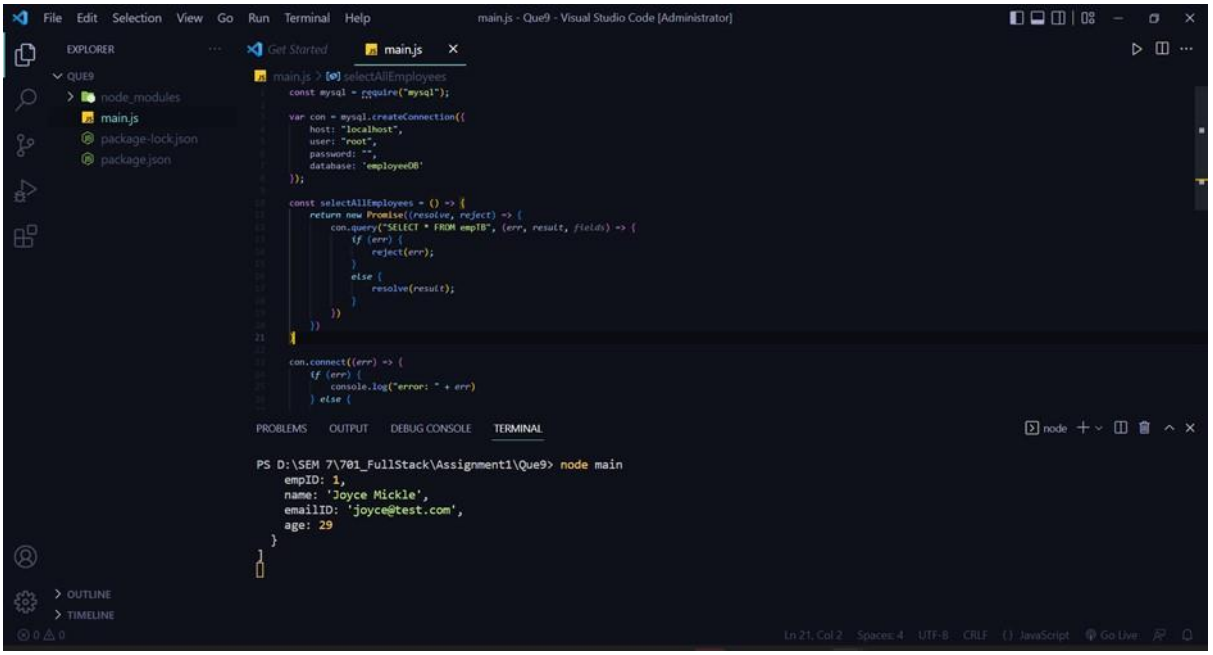
Q:9. Write a program that connect Mysql database, Insert a record in employee table and display all records in employee table using promise based approach.

CODE:

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


var con = mysql.createConnection({
    host: "localhost",
    user: "root",
    password: "",
    database: 'employeeDB'
});


const selectAllEmployees = () => {
    return new Promise((resolve, reject) => {
        con.query("SELECT * FROM empTB", (err, result, fields) => {
            if (err) {
                reject(err);
            } else
            {
                resolve(result);
```

```
            }
        })
    })
}


con.connect((err) => {
    if (err) {
        console.log("error: " + err)
    } else {

        //inserting record in employee table
        con.query("INSERT INTO empTB values(null,'Joyce
Mickle','joyce@test.com',29)", (err, result) => {
            if (err) {
                console.log("error: " + err)
            } else {
                console.log("record inserted")
            }
        })

        //getting all data from employee table using promise based
approach
        selectAllEmployees().then(result => {
            console.log(result)
        }).catch(err => {
            console.log("error: " + err)
        })
    }
})
```

OUTPUT:

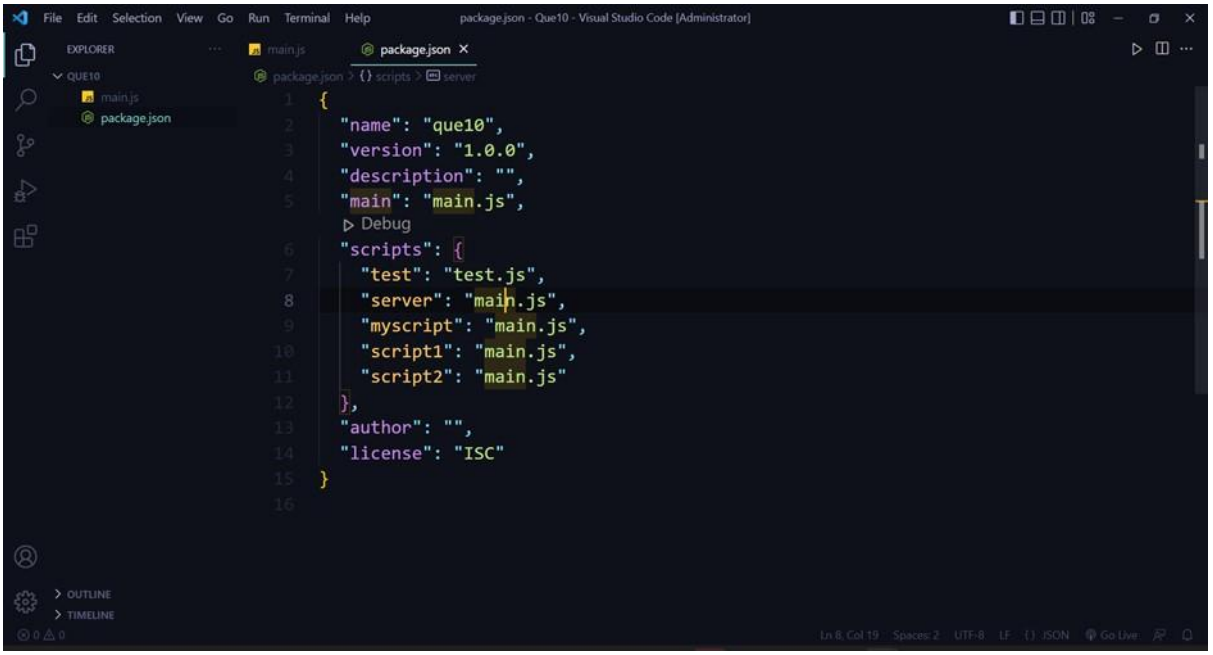


Q:10. Set a server script, a test script and 3 user defined scripts in package.json file in your nodejs application.

CODE:

```
console.log("Hello From Question 10!");
```

OUTPUT:



Q:11. Develop an application to show live cricket score.

CODE:

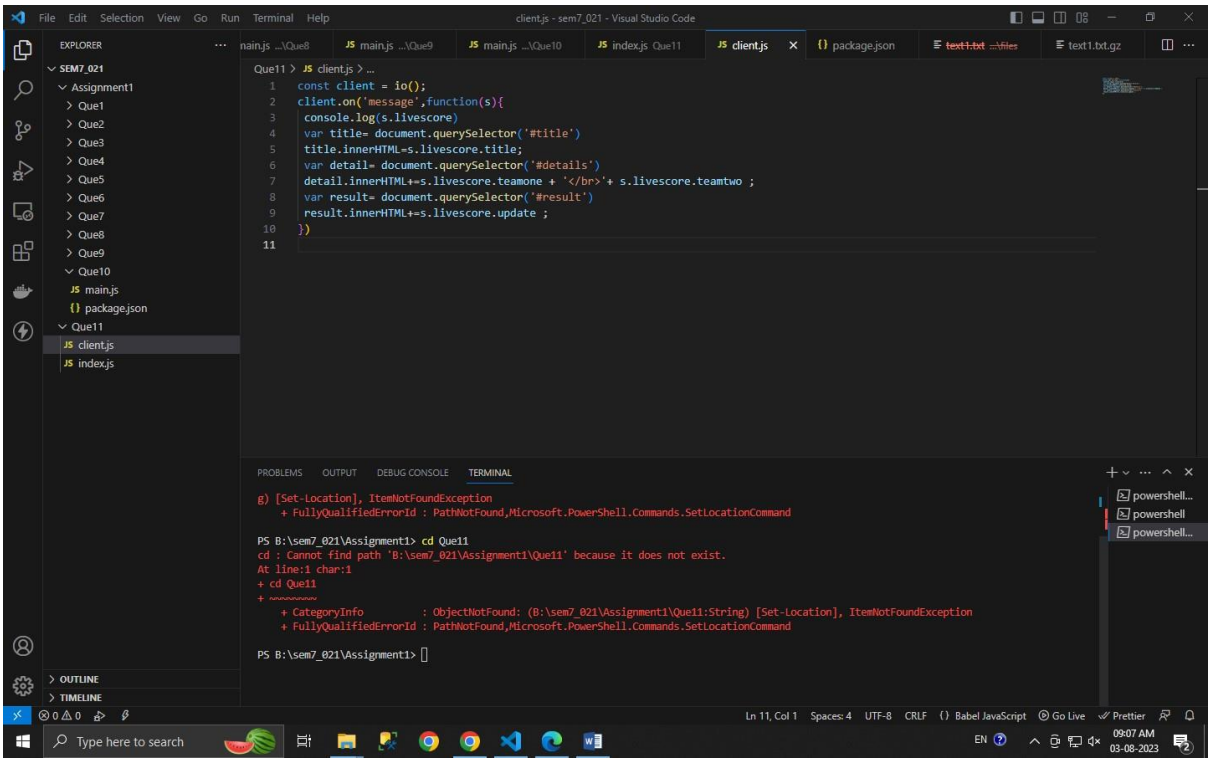
```
const express = require('express');
const app = express();
```

```
const http = require('http').createServer(app)
const PORT = process.env.PORT || 5000
var data;
const socket = require('socket.io')(http)
const fetch=require('node-fetch')
var
data=fetch("https://cricket-
api.vercel.app/cri.php?url=https://www.cric
buzz.com/live-cricket-scores/32057/3rd-test-india-tour-of-england-
2022"
).then(res => res.json())
.then(json => {
 data=json
});
socket.on('connection',function(s){
 console.log("Helloo "+ data)
 s.emit('message',data);


})
http.listen(PORT,() => {
 console.log(`Listening on port ${PORT}`);
})
app.use(express.static(__dirname+'/public'))
app.get('/',(req,res) => {
 res.sendFile(__dirname + '/index.html')
})
```

Client.js

```
const client = io();
client.on('message',function(s){
 console.log(s.livescore)
 var title= document.querySelector('#title')
 title.innerHTML=s.livescore.title;
 var detail= document.querySelector('#details')
 detail.innerHTML+=s.livescore.teamone + '</br>'+ s.livescore.teamtwo
 ;
 var result= document.querySelector('#result')
 result.innerHTML+=s.livescore.update ;
})
```

OUTPUT:


File Edit Selection View Go Run Terminal Help
client.js - sem7_021 - Visual Studio Code
EXPLORER
SEM7_021
Assignment1
Que1
Que2
Que3
Que4
Que5
Que6
Que7
Que8
Que9
Que10
main.js
package.json
Que11
client.js
index.js
OUTLINE
TIMELINE
Que11 > client.js > ...
1 const client = io();
2 client.on('message',function(s){
3 console.log(s.livescore)
4 var title= document.querySelector('#title')
5 title.innerHTML=s.livescore.title;
6 var detail= document.querySelector('#details')
7 detail.innerHTML+=s.livescore.teamone + '</br>'+ s.livescore.teamtwo ;
8 var result= document.querySelector('#result')
9 result.innerHTML+=s.livescore.update ;
10 })
11
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
g) [Set-Location], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand
PS B:\sem7_021\Assignment1> cd Que11
cd : Cannot find path 'B:\sem7_021\Assignment1\Que11' because it does not exist.
At line:1 char:1
+ cd Que11
+ ~~~~~~~~
+ CategoryInfo : ObjectNotFound: (B:\sem7_021\Assignment1\Que11:String) [Set-Location], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand
PS B:\sem7_021\Assignment1>
powershell...
powershell
powershell...
Ln 11, Col 1 Spaces: 4 UTF-8 CRLF Babel JavaScript Go Live Prettier
Type here to search
EN
09:07 AM
03-08-2023