**Name :** Rana Dhruvi Ajaybhai

**Roll No. :** 068

**Division :** A

**Semester:** 7rd

**Subject :** 705 Full Stack Development

**Date :** 30/07/2023

**GitHub Link :** <https://github.com/DhruviRana4/705_Assignment1_068>

**PRACTICAL ASSIGNMENT: 1**

**Question: 1**

Develop a web server with following functionalities:

- Serve static resources.

- Handle GET request.

- Handle POST request.

**Answer: 1**

const http=require("http");

const static=require("node-static");

var url=require("url");

var fileServer=new static.Server("testFolder");

const server=http.createServer((req,res)=>{

    console.log("Hello node...!!");

    fileServer.serve(req,res);

    var url2=url.parse(req.url,true);

    console.log(url2.pathname)

    if(url2.pathname=="/"){

        fileserver.serve(req,res)

    }else if(url2.pathname=="/process" && req.method==="GET"){

        res.end("Email : " + url2.query.user\_email  + " Password : " + url2.query.pwd);

    }else if(url2.pathname=="/process\_post" && req.method==="POST"){

        // console.log("first")

        let body="";

        req.on("data",chunk=>{

            body+=chunk;

        });

        req.on("end",()=>{

            res.write(body);

            res.end();

        })

    }else{

        res.end("page not found...!!")

    }

})

server.listen(8080,()=>{

    console.log("Server started...!!")

})

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

    integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

</head>

<body>

  <div class="container w-50 mt-5">

    <img src="Images/wallpaperflare.com\_wallpaper (2).jpg" alt="" class="w-25 h-25">

    <h1>Form with Post method</h1>

    <form method="GET" action="/process">

      <div class="form-group">

        <label for="exampleInputEmail1">Email address</label>

        <input type="email" name="user\_email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp"

          placeholder="Enter email">

        <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

      </div>

      <div class="form-group">

        <label for="exampleInputPassword1">Password</label>

        <input type="password" name="pwd" class="form-control" id="exampleInputPassword1" placeholder="Password">

      </div>

      <button type="submit" class="btn btn-primary">Submit</button>

    </form>

    <hr>

    <hr>

    <h1>Form with Post method</h1>

    <form method="POST" action="/process\_post">

      <div class="form-group">

        <label for="exampleInputEmail1">Email address</label>

        <input type="email" name="user\_email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp"

          placeholder="Enter email">

        <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

      </div>

      <div class="form-group">

        <label for="exampleInputPassword1">Password</label>

        <input type="password" name="pwd" class="form-control" id="exampleInputPassword1" placeholder="Password">

      </div>

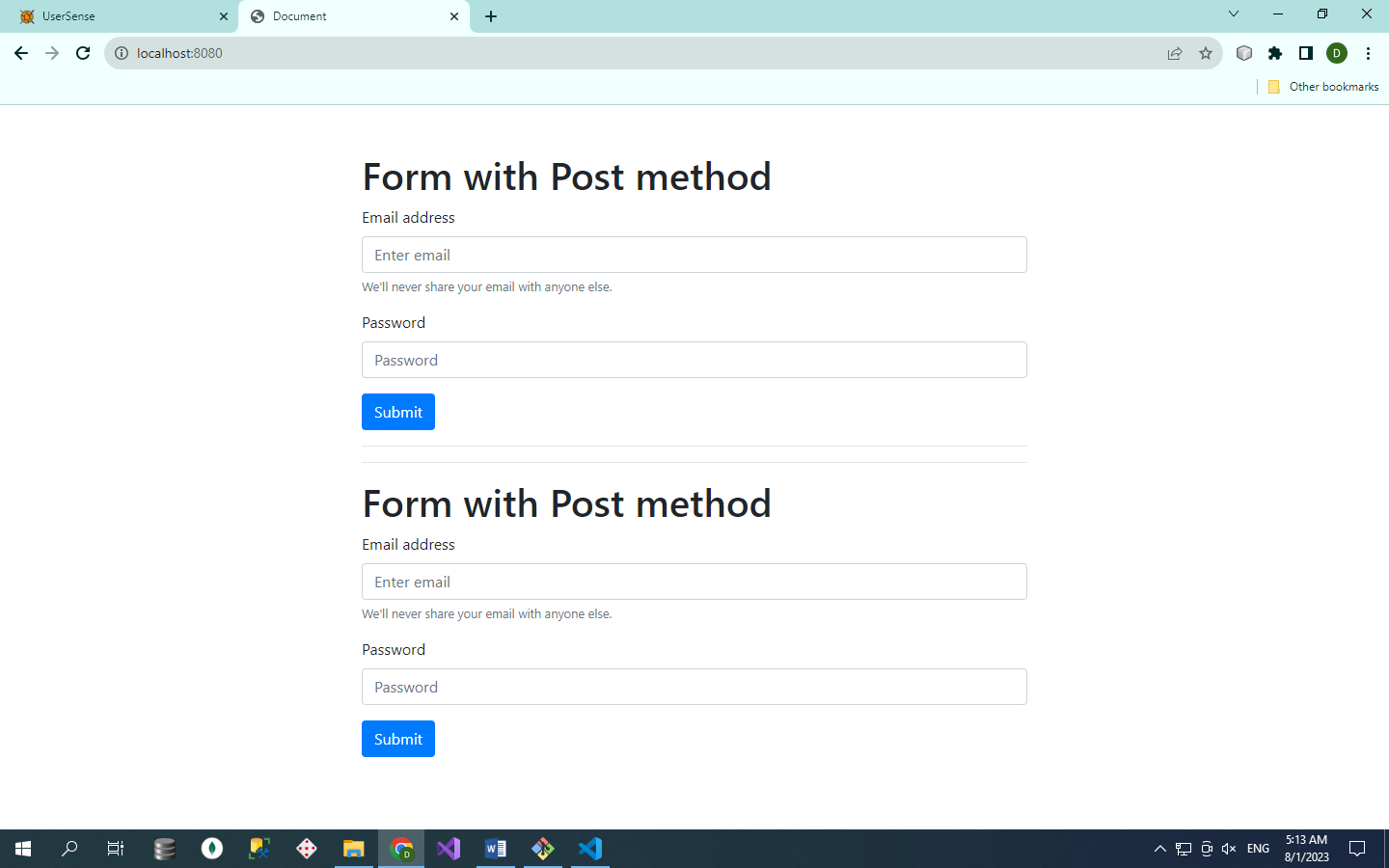
      <button type="submit" class="btn btn-primary">Submit</button>

    </form>

  </div>

</body>

</html>



**Question: 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.)

**Answer: 2**

const http = require("http");

const static = require("node-static");

const url = require("url");

var fileserver = new static.Server("./staticFiles");

const server = http.createServer((req, res) => {

    console.log("Hello NodeJS...!!");

    var url1 = url.parse(req.url, true);

    if (url1.pathname == "/getHello" && req.method == "GET") {

        res.end("Hello NodeJs...!!");

    }

    else if (url1.pathname === '/') {

        fileserver.serve(req, res);

    }

    else {

        res.writeHead(404, { 'Content-Type': 'text/plain' });

        res.end('Page not found');

    }

})

server.listen(8080, () => {

    console.log("Server listening on port 8000..!!");

})

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"

        integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

</head>

<body>

    <div class="container d-flex justify-content-center " style="height: 100vh;">

        <div class="align-self-center">

                <button class="btn btn-secondary" id="clickBtn" name="Submit">Click Here</button><br>

                <label for="" class="mt-2" id="result"></label>

        </div>

    </div>

</body>

<script>

    document.getElementById("clickBtn").addEventListener("click",async()=>{

        var response =await fetch("http://localhost:8080/getHello");

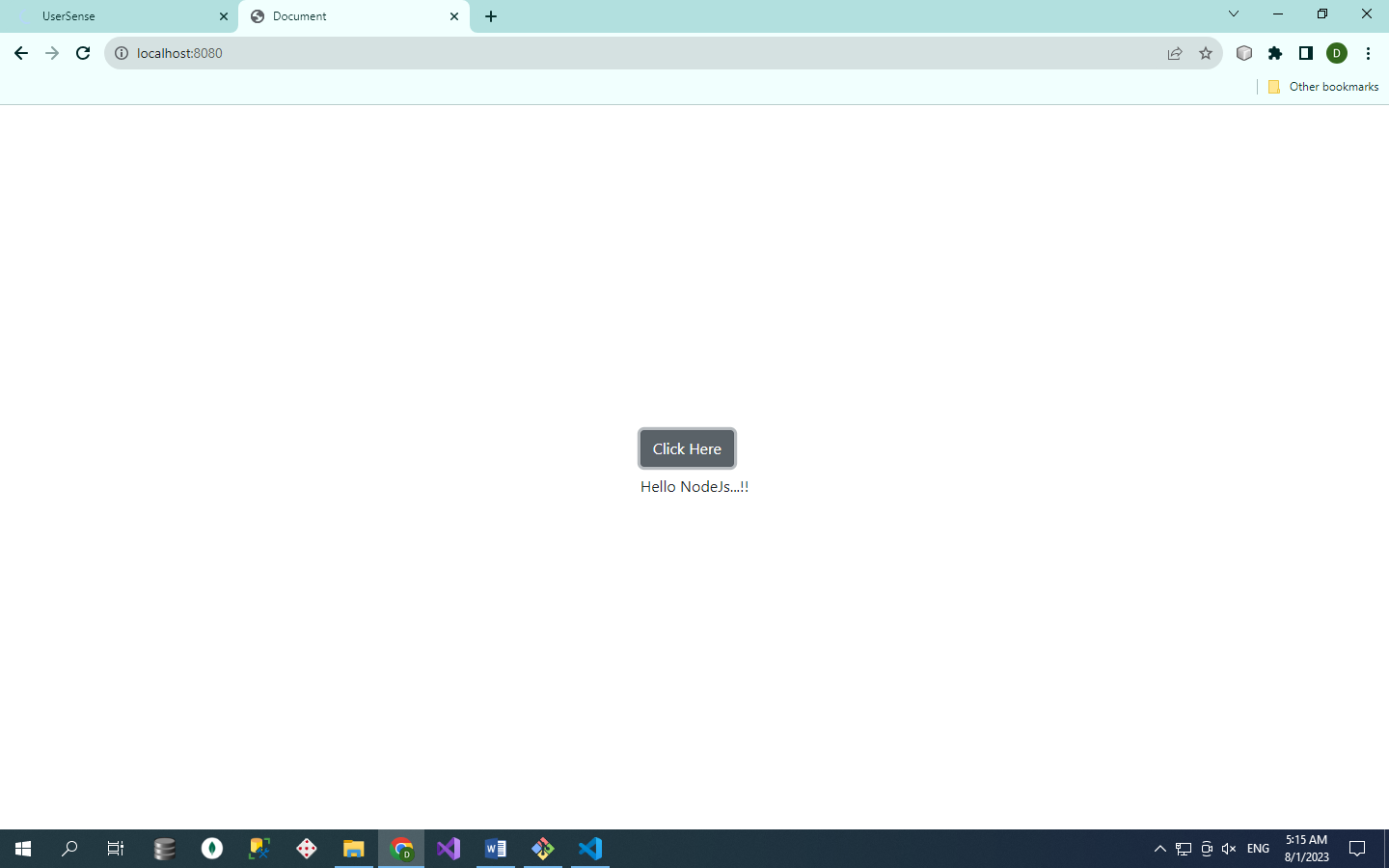
        var text=await response.text();

        document.getElementById("result").textContent=text;

    })

</script>

</html>



**Question: 3**

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

**Answer: 3**

var Chatbot = require('./chatBoat');

var readline = require('readline');

var r1 = readline.createInterface(process.stdin, process.stdout);

r1.setPrompt("You==>");

r1.prompt();

r1.on('line', function(message) {

    console.log('Bot ==> '+ Chatbot.reply(message));

    r1.prompt();

}).on('close',function(){  //chaining events.

    process.exit(0);

});

module.exports.reply = function (msg) {

    this.Boat\_age = 19;

    this.Boat\_Name = "Dhruvi";

    this.Bot\_Country = "India";

    msg = msg.toLowerCase();

    if(msg.indexOf("hi") > -1 ||

            msg.indexOf("hello") > -1 ||

            msg.indexOf("welcome") > -1 )

        {

            return "Hi!";

        }

        else if(msg.indexOf("age") > -1 &&

            msg.indexOf("your"))

        {

            return "I'm " + this.Bot\_Age;

        }

        else if (msg.indexOf("how") > -1 &&

            msg.indexOf("are") &&

            msg.indexOf("you"))

        {

            return "I'm fine ^\_^"

        }

        else if(msg.indexOf("where") > -1

            && msg.indexOf("live") &&

            msg.indexOf("you"))

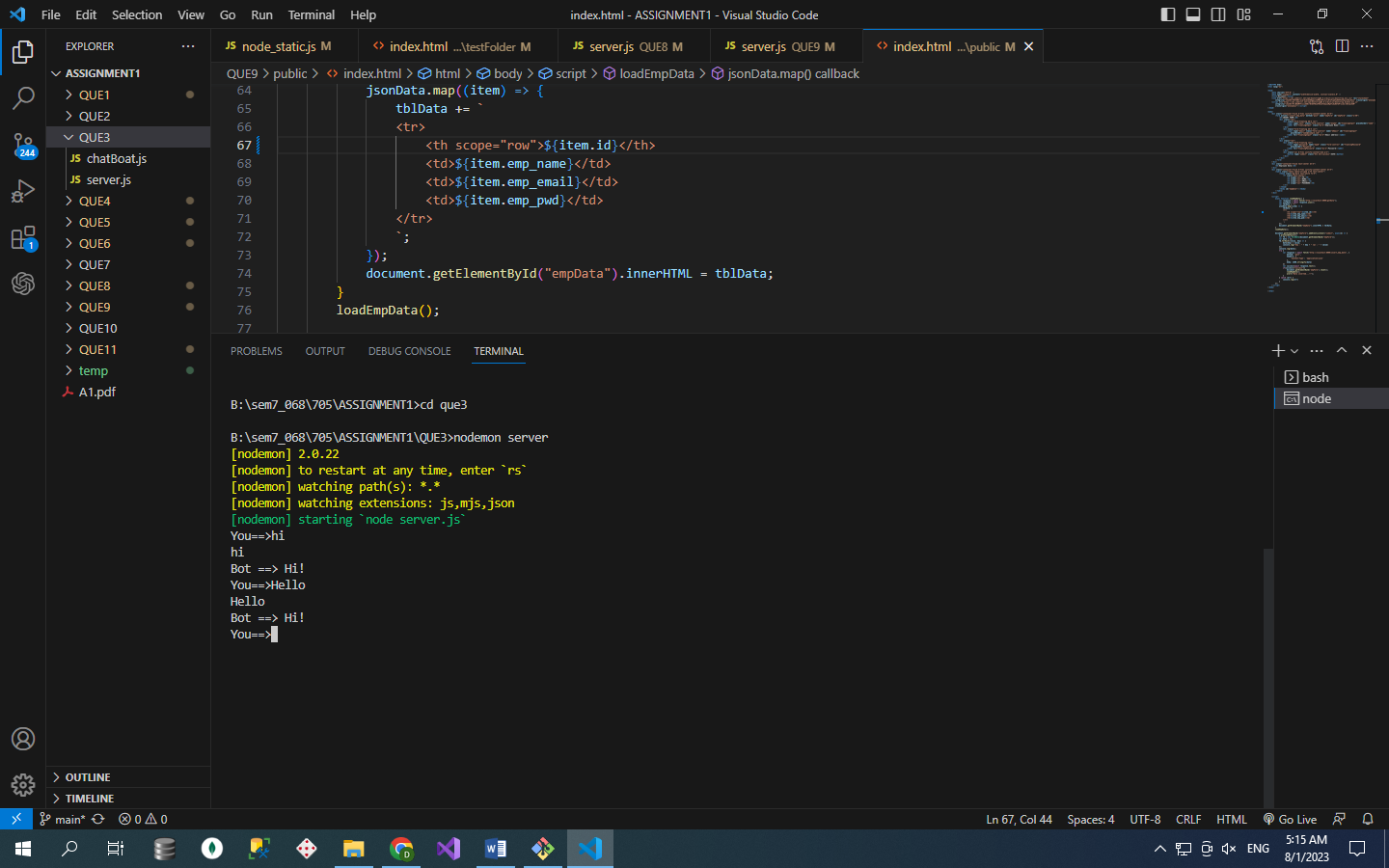
        {

            return "I live in " + this.Bot\_Country;

        }

        return "Sorry, I didn't get it :( ";

}



**Question: 4**

Use above chatbot module in web based chatting of websocket.

**Answer: 4**

const http=require("http");

const static=require("node-static");

const url=require("url");

const websocket=require("ws");

var fileServer=new static.Server("./public");

var server=http.createServer((req,res)=>{

    var url2=url.parse(req.url,true);

    // console.log("Hello NodeJs...!!");

    if(url2.pathname==""){

        console.log(url2)

    }

    fileServer.serve(req,res);

})

server.listen(8000,()=>{

    console.log("server listening on port 8000");

})

var wss=new websocket.Server({server:server});

wss.on("connection",(ws)=>{

    ws.send("hello client..!!");

    ws.on("message",(msg)=>{

        ws.send("I recieved ==> " + msg);

    })

})

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"

        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous" />

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"

        crossorigin="anonymous"></script>

</head>

<body>

    <div></div>

    <div class="container-fluid border w-100 d-flex p-5">

        <div class="col">

            <h3>Client</h3>

            <form class="form d-flex w-50" id="clientForm">

                <input type="text" class="form-control pe-3" id="clientMsg" />

                <input type="submit" class="btn btn-success ms-3" value="Send" id="sendMsg" name="sendMsg" />

            </form>

        </div>

        <div class="col">

            <h3>Server</h3>

            <div class="container-fuild" id="chat\_data"></div>

        </div>

    </div>

    <script>

        var wss = new WebSocket("ws://localhost:8000");

        wss.addEventListener("message", (e) => {

            var msg = e.data;

            document.getElementById("chat\_data").innerHTML +=

                "<b>Server</b> : " + msg + "</br>";

        });

        function senDataToServer(){

            var clientMsg = document.getElementById("clientMsg").value;

            wss.send(clientMsg);

            document.getElementById("chat\_data").innerHTML +=

                "<b>Client</b> : " + clientMsg + "</br>";

            document.getElementById("clientMsg").value = "";

        }

        document.getElementById("sendMsg").addEventListener("click", (e) => {

            e.preventDefault();

            senDataToServer();

        });

        document.getElementById("clientForm").addEventListener("submit",(e)=>{

            e.preventDefault();

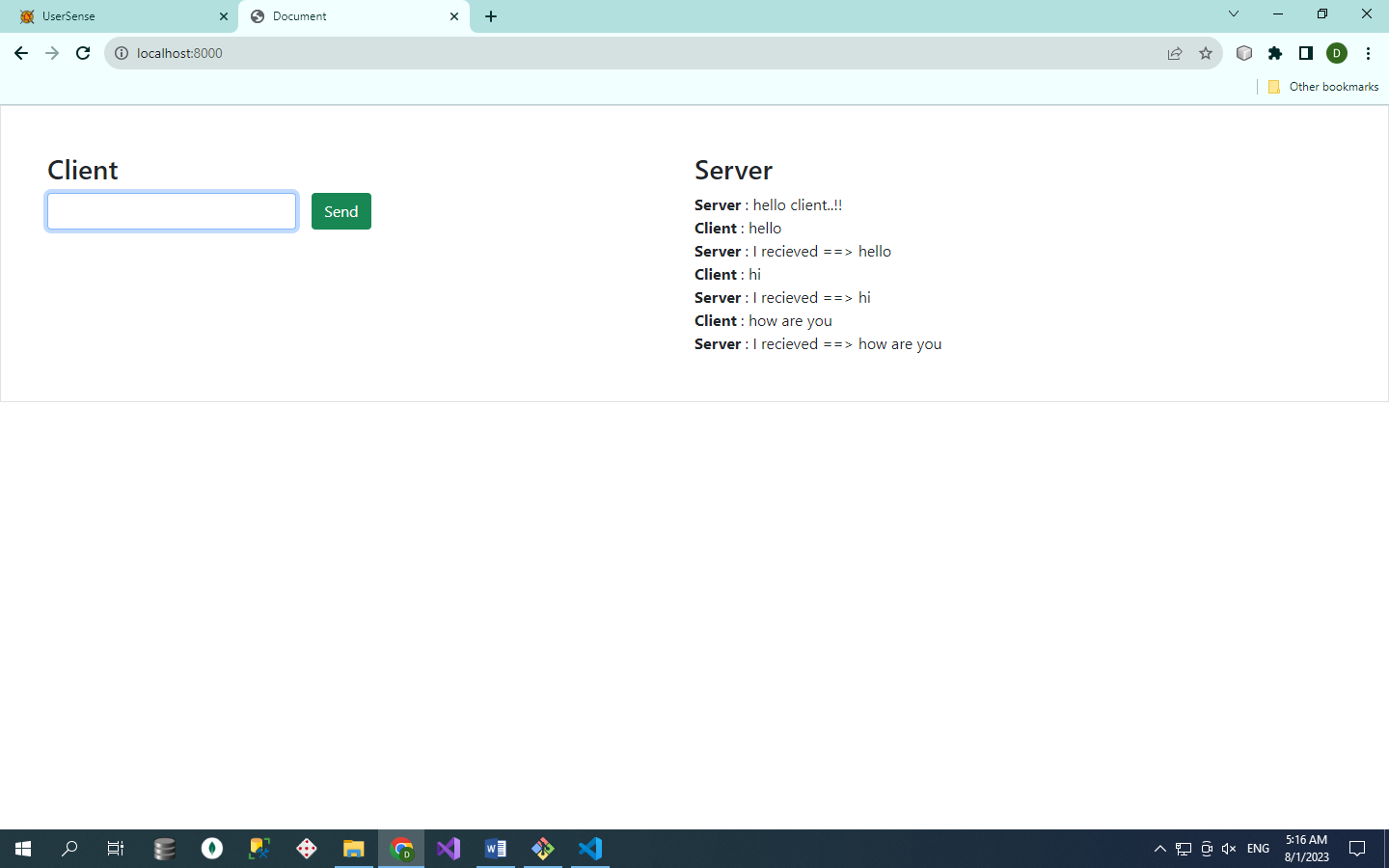
            senDataToServer();

        })

    </script>

</body>

</html>



**Question: 5**

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

**Answer: 5**

const fs = require('fs');

const archiver = require('archiver');

function createZipFolder(folderPath, zipFilePath) {

    const output = fs.createWriteStream(zipFilePath);

    const archive = archiver('zip', {

        zlib: { level: 9 }

    });

    output.on('close', () => {

        console.log(`Successfully created ${zipFilePath}`);

    });

    archive.on('error', (err) => {

        throw err;

    });

    archive.pipe(output);

    archive.directory(folderPath, false);

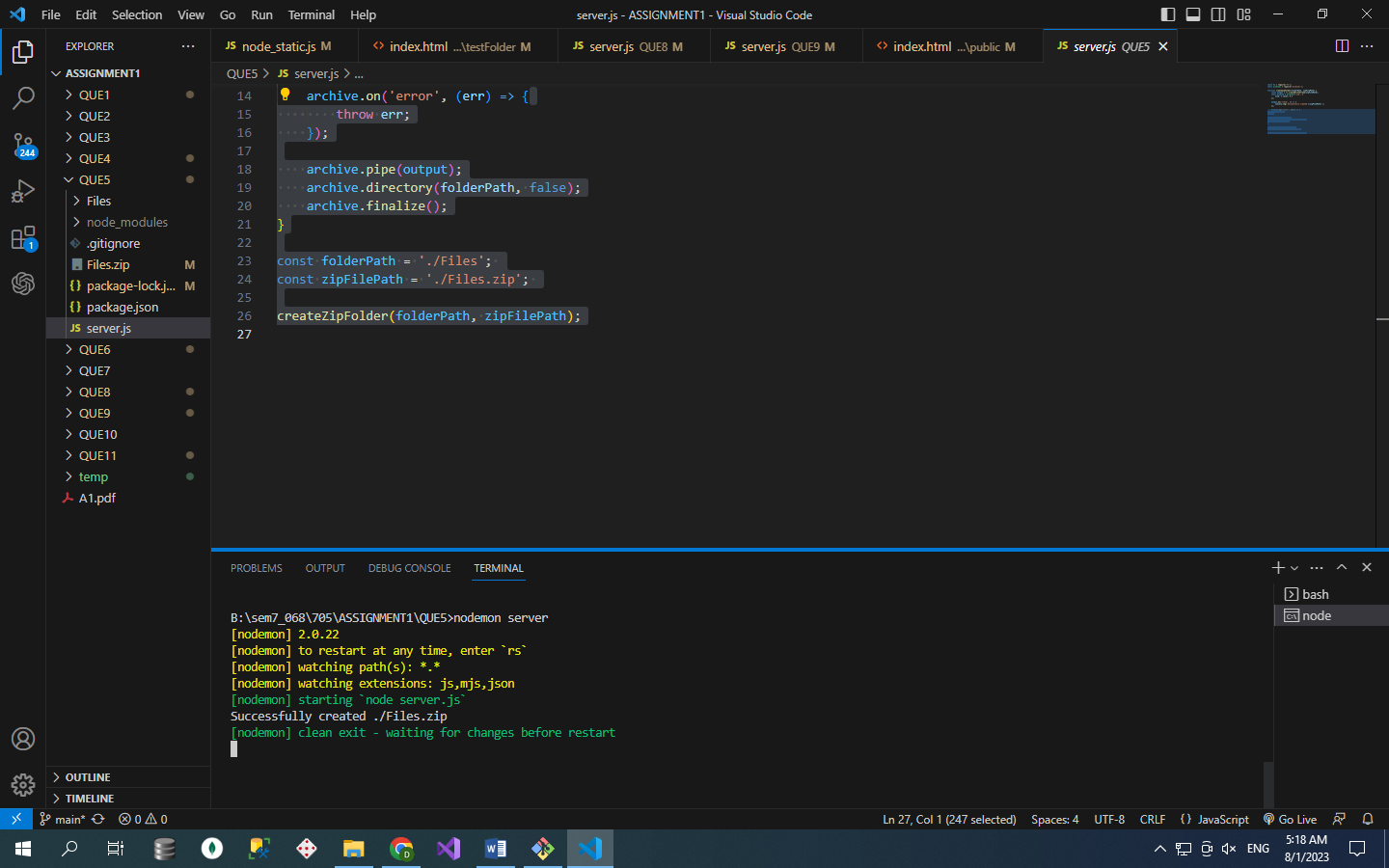
    archive.finalize();

}

const folderPath = './Files';

const zipFilePath = './Files.zip';

createZipFolder(folderPath, zipFilePath);



**Question: 6**

Write a program to extract a zip file.

**Answer: 6**

const fs = require('fs');

const unzipper = require('unzipper');

async function extractZip(zipFilePath, extractToPath) {

    try {

        const stream = fs.createReadStream(zipFilePath);

        await stream.pipe(unzipper.Extract({ path: extractToPath }));

        console.log('Extraction complete.');

    } catch (err) {

        console.error('Error extracting the zip file:', err);

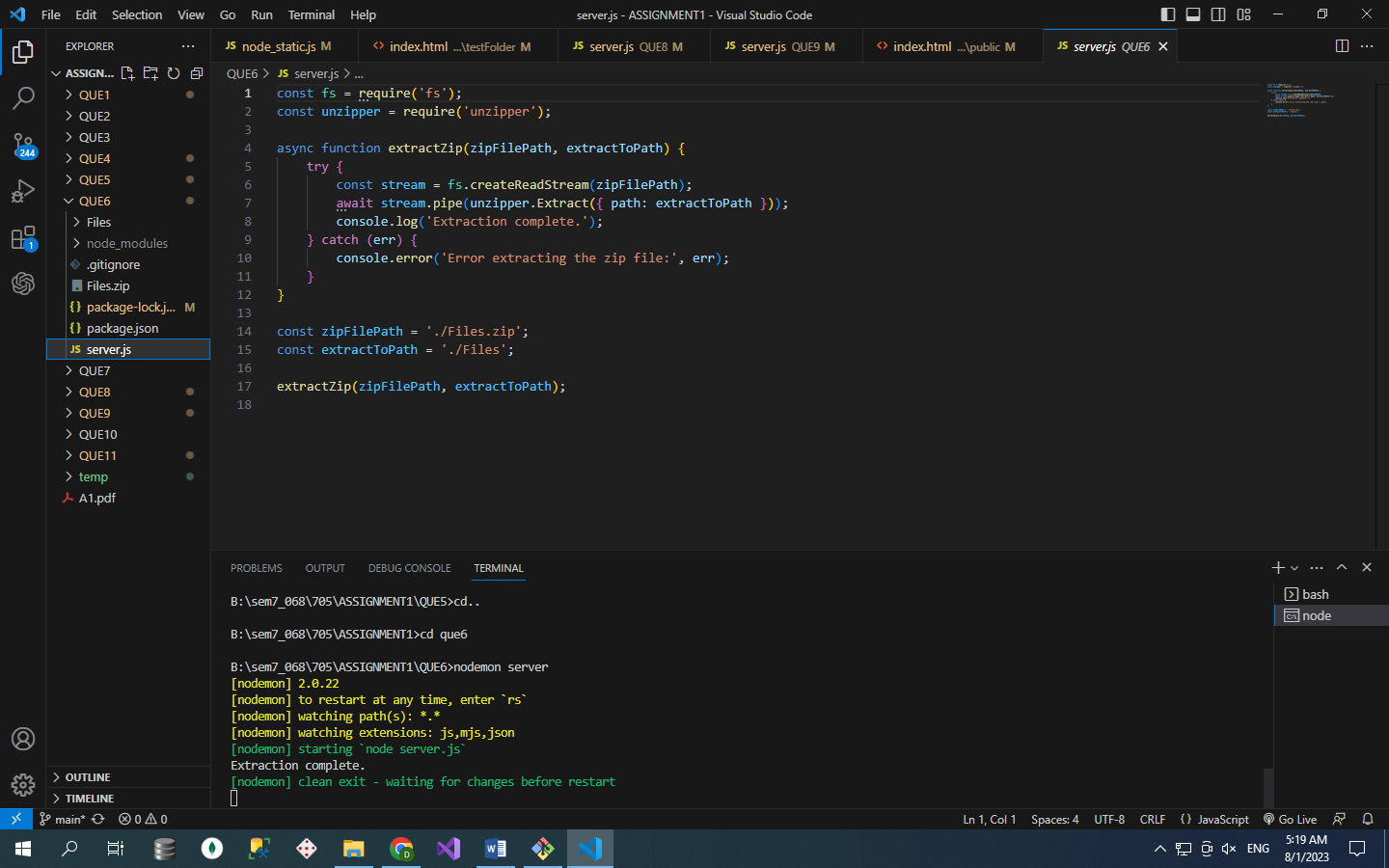
    }

}

const zipFilePath = './Files.zip';

const extractToPath = './Files';

extractZip(zipFilePath, extractToPath);



**Question: 7**

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

**Answer: 7**

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.')

            }

        })

    })

}

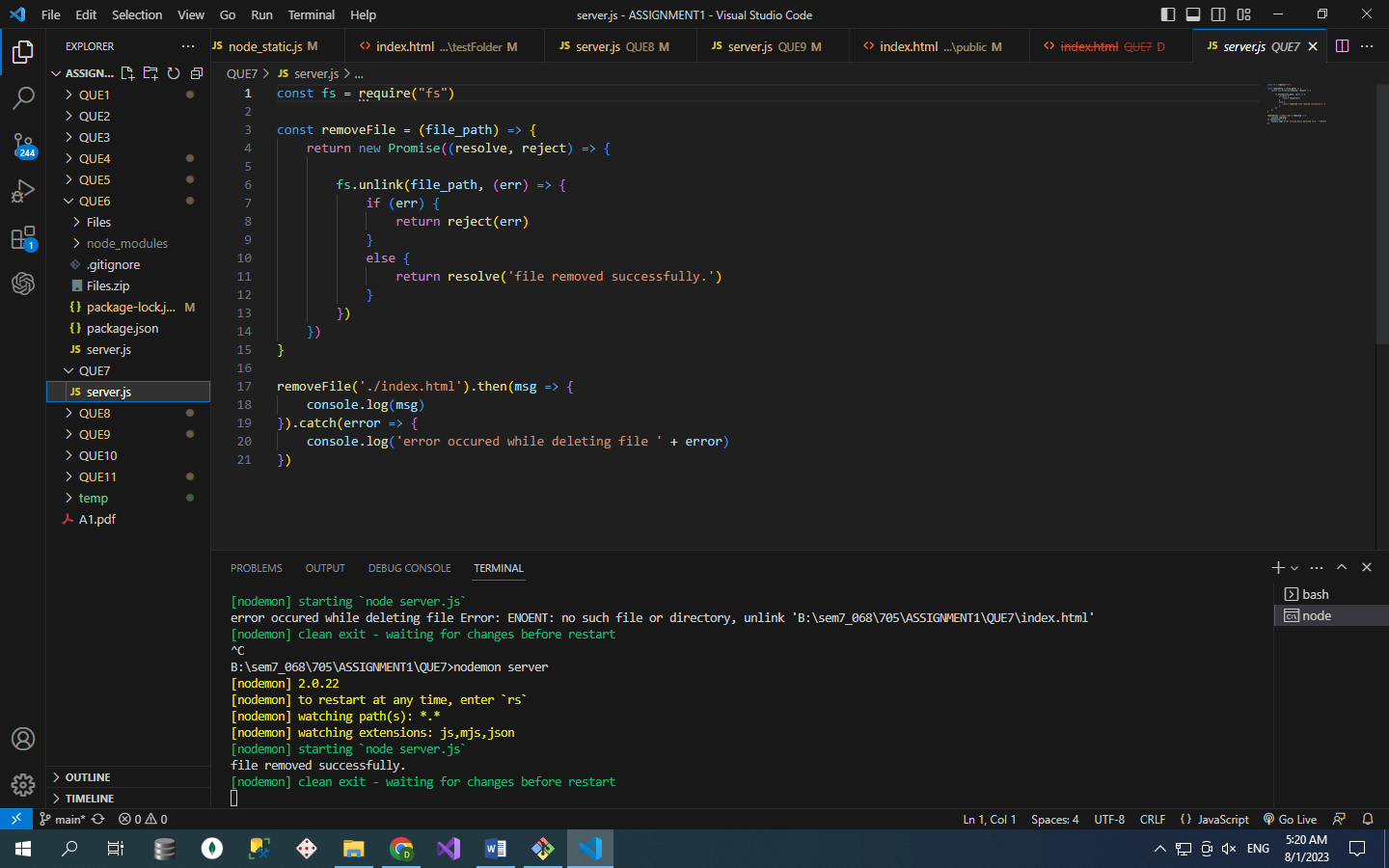
removeFile('./index.html').then(msg => {

    console.log(msg)

}).catch(error => {

    console.log('error occured while deleting file ' + error)

})



**Question: 8**

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

**Answer: 8**

(async () => {

    try {

      const response = await fetch("https://www.google.com/");

      const text = await response.text();

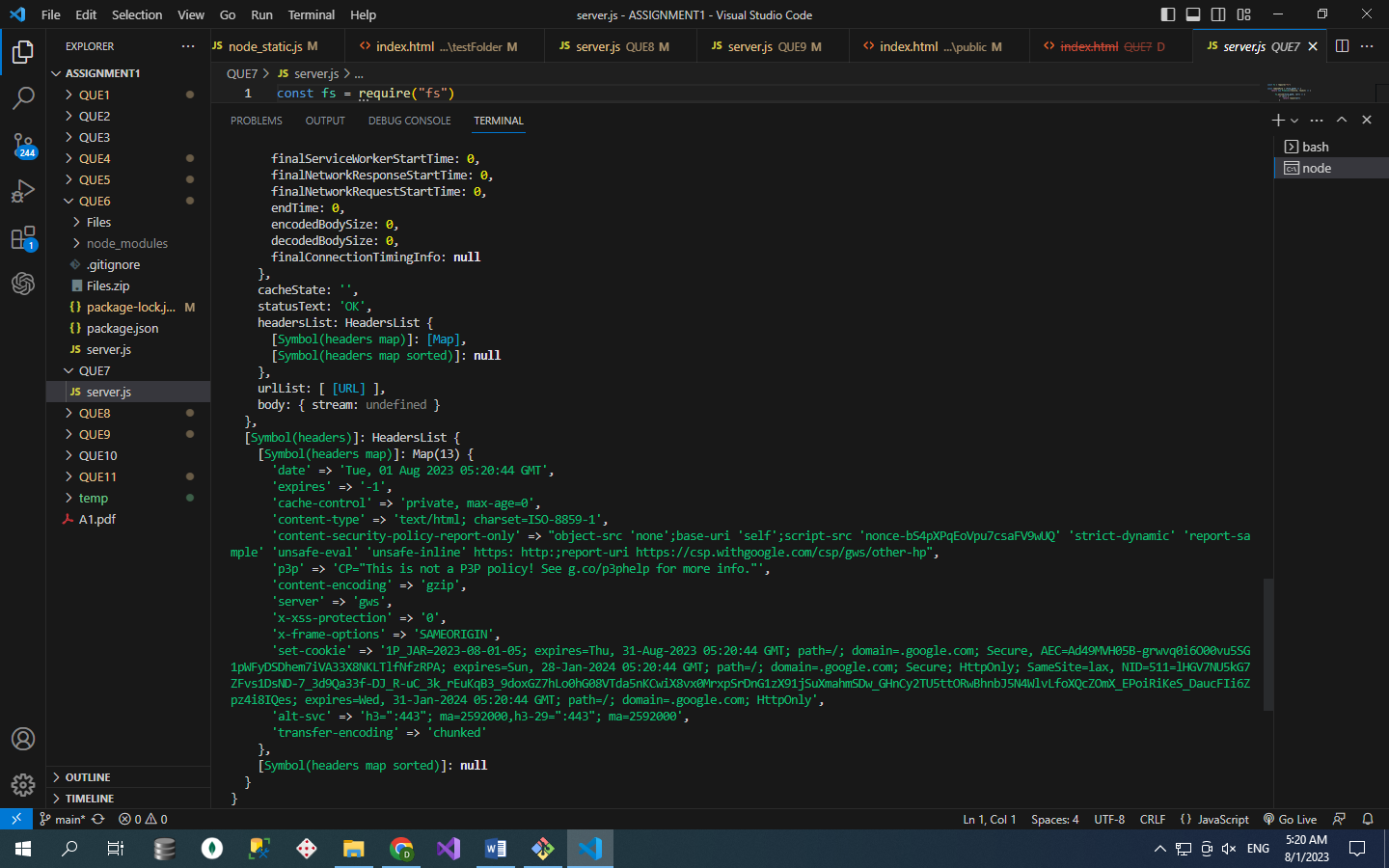
      console.log(text);

    } catch (error) {

      console.log(error.response.body);

    }

  })();



**Question: 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.

**Answer: 9**

const http=require("http");

const mysql=require("mysql");

const static=require("node-static");

var fileserver=new static.Server("./public");

var conn=mysql.createConnection({

    host:"localhost",

    user:"root",

    password:"",

    database:"employeedb"

});

conn.connect((err)=>{

    if(err){

        console.log(err);

    }else{

        console.log("connected")

    }

})

async function getData(){

}

var server=http.createServer((req,res)=>{

    console.log(req.url);

    if(req.url=="/"){

        fileserver.serve(req,res);

    }

    if(req.url=="/getData"){

        conn.query("SELECT \* FROM `emptb`",(err,data)=>{

            if(err){

                return "err";

            }

            res.end(JSON.stringify(data));

        })

    }

    if(req.url=="/insert\_emp\_data" && req.method==="POST"){

        let data = '';

        req.on('data', (chunk) => {

            data += chunk;

        });

        req.on("end",()=>{

            var fd=JSON.parse(data);

            // console.log(fd.name)

            var sql=`INSERT INTO emptb(emp\_name, emp\_email, emp\_pwd) VALUES ('${fd.ename}','${fd.eEmail}','${fd.epwd}')`;

            conn.query(sql,(err,data)=>{

                if(err){

                    console.log(err);

                }else{

                    res.end("success");

                }

            })

        })

        // res.end();

    }

})

server.listen(8000,()=>{

    console.log("server listening on port 8000");

})

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"

        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous" />

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"

        crossorigin="anonymous"></script>

</head>

<body>

    <div class="container-fluid d-flex justify-content-center mt-5">

        <form action="/insert\_emp\_data" method="post" name="empForm" id="empForm" class="w-50">

            <h4>INSERT FORM</h4>

            <div class="row">

                <div class="form-floating mb-3 col">

                    <input type="text" class="form-control" name="ename" id="floatingInput" placeholder="name" />

                    <label for="floatingInput" class="ms-2">Employee Name</label>

                </div>

                <div class="form-floating mb-3 col">

                    <input type="email" class="form-control" name="eEmail" id="floatingInput"

                        placeholder="name@example.com" />

                    <label for="floatingInput" class="ms-2">Email address</label>

                </div>

            </div>

            <div class="row">

                <div class="form-floating col">

                    <input type="password" name="epwd" class="form-control" id="floatingPassword"

                        placeholder="Password" />

                    <label for="floatingPassword" class="ms-2">Password</label>

                </div>

                <div class="col d-flex justify-content-end p-2">

                    <button type="submit" class="btn btn-success">SAVE</button>

                </div>

            </div>

        </form>

    </div>

    <div class="container-fluid text-center mt-3">

        <h3>Employee Data</h3>

    </div>

    <div class="container-fluid d-flex justify-content-center mt-4">

        <table class="table table-striped w-75 text-center">

            <thead class="bg-dark text-white fs-5">

                <tr class="text-center">

                    <th scope="col">ID</th>

                    <th scope="col">NAME</th>

                    <th scope="col">EMAIL</th>

                    <th scope="col">PASSWORD</th>

                </tr>

            </thead>

            <tbody id="empData"></tbody>

        </table>

    </div>

    <script>

        async function loadEmpData() {

            var response = await fetch("http://localhost:8000/getData");

            var jsonData = await response.json();

            var tblData = "";

            jsonData.map((item) => {

                tblData += `

                <tr>

                    <th scope="row">${item.emp\_id}</th>

                    <td>${item.emp\_name}</td>

                    <td>${item.emp\_email}</td>

                    <td>${item.emp\_pwd}</td>

                </tr>

                `;

            });

            document.getElementById("empData").innerHTML = tblData;

        }

        loadEmpData();

        document.getElementById("empForm").addEventListener("submit", async(e) => {

            e.preventDefault();

            var fd = new FormData(document.getElementById("empForm"));

            var data = {};

            fd.forEach((value, key) => {

                data[key] = value;

                console.log("key : " + key + " val : " + value)

            });

            console.log(data);

            try {

                var response = await fetch("http://localhost:8000/insert\_emp\_data", {

                    method: 'post',

                    headers: {

                        'Content-Type': 'application/json'

                    },

                    body: JSON.stringify(data)

                })

                var success=await response.text();

                if(success=="success"){

                    document.getElementById('empForm').reset();

                    loadEmpData();

                    alert("Data inserted...!!");

                }

            } catch (err) {

                console.log(err)

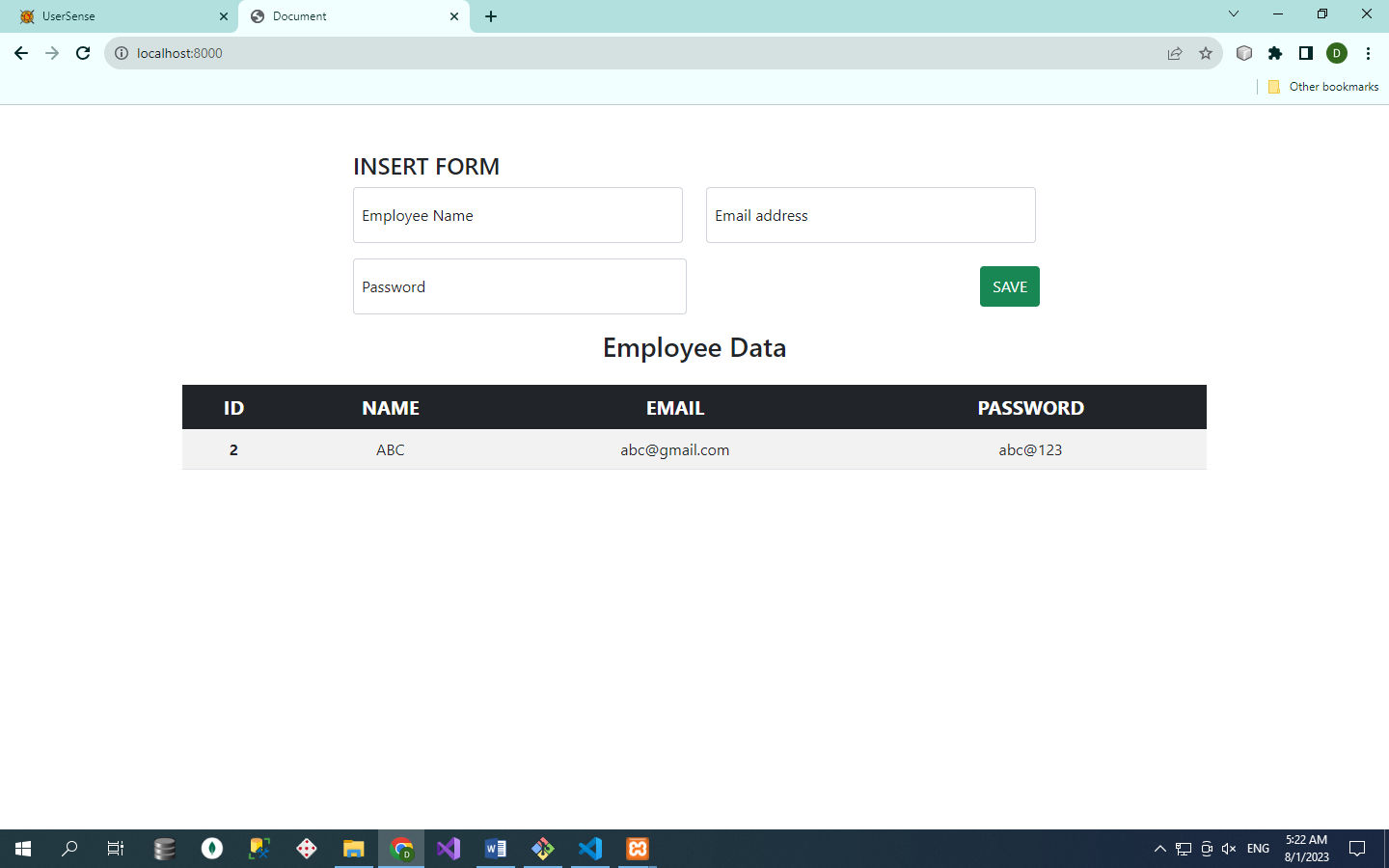
            }

        });

    </script>

</body>

</html>



**Question: 10**

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

**Answer: 10**

{

  "name": "que10",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1",

    "server1":"nodemon server1.js",

    "server2":"nodemon server2.js",

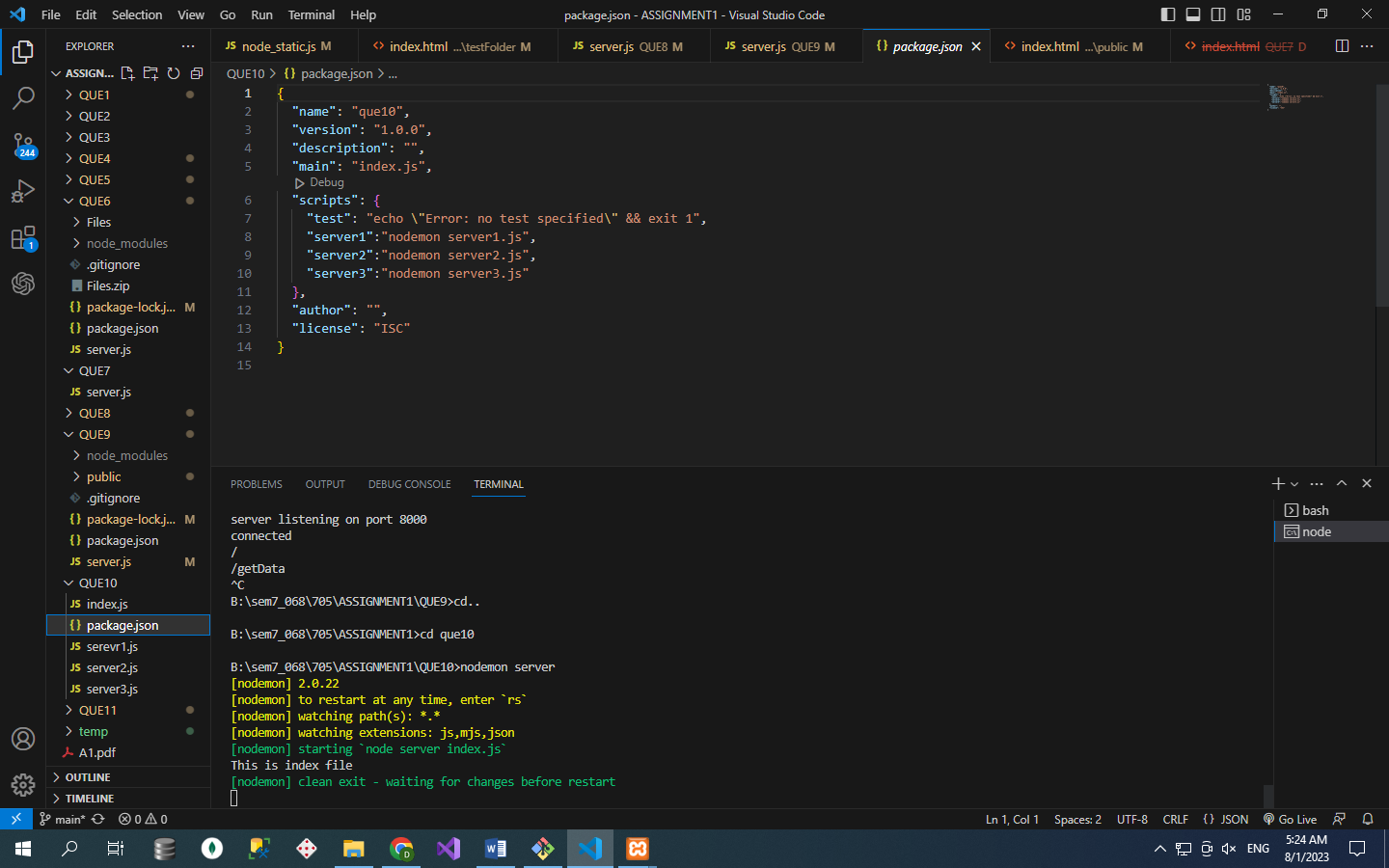
    "server3":"nodemon server3.js"

  },

  "author": "",

  "license": "ISC"

}



**Question: 11**

Develop an application to show live cricket score.

**Answer: 11**

const axios = require("axios");

const http = require("http");

const static = require("node-static");

const url = require("url");

const websocket = require("ws");

var fileServer = new static.Server("./public");

var server = http.createServer((req, res) => {

    fileServer.serve(req, res);

})

var latestData = null;

server.listen(8000, () => {

    console.log("server listening on port 8000");

})

async function fetchMatchScore() {

    try {

        var response = await axios.get("https://api.cricapi.com/v1/currentMatches?apikey=0bf9e0f5-5333-4925-912f-5a5511d62c19&offset=0");

        return response.data;

    } catch (err) {

        console.log(err)

    }

}

var wss = new websocket.Server({ server: server });

wss.on("connection", async (ws) => {

    var data = await fetchMatchScore();

    ws.send(JSON.stringify(data));

})

async function updateDataAndBroadcast() {

    latestData = await fetchMatchScore();

    if (latestData !== null) {

        wss.clients.forEach((client) => {

            if (client.readyState === websocket.OPEN) {

                client.send(JSON.stringify(latestData));

            }

        });

    }

}

setInterval(updateDataAndBroadcast, 5000);

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <title>Document</title>

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"

    integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous" />

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

    integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"

    crossorigin="anonymous"></script>

</head>

<body>

  <div></div>

  <div class="container-fluid border w-100 d-flex p-5">

    <div class="col">

      <h3>Live Cricket Score</h3>

      <div class="container-fuild" id="chat\_data"></div>

    </div>

  </div>

  <script>

    var wss = new WebSocket("ws://localhost:8000");

    wss.addEventListener("message", (e) => {

      var msg = JSON.parse(e.data);

      console.log(msg);

      var scoreData = "<div class='row'>";

      msg.data.map((item) => {

        scoreData += `

                <div class="col-sm-6 mb-4">

                    <div class="card">

                      <div class="card-body">

                          <h5 class="card-title">${item.name}</h5>

                          <p class="card-text">${item.date}</p>

                          <div class='row'>

                          `;

        item.score.map((val) => {

          console.log(val)

          scoreData += `<div class='col-6'>

                            <a class="fw-bold text-decoration-none text-dark mb-2 fs-6">Inning : ${val.inning}</a></br>

                            <a class="btn btn-outline-primary mb-1">Over : ${val.o}</a></br>

                            <a class="btn btn-outline-success mb-1">Run : ${val.r}</a></br>

                            <a class="btn btn-outline-danger mb-1">Wicket : ${val.w}</a></br>

                          </div>`;

        });

        scoreData += `

          </div></div>

                    </div>

                </div>`;

        // console.log(item.name);

      });

      scoreData += `</div>`;

      document.getElementById("chat\_data").innerHTML = scoreData + "</br>";

    });

  </script>

</body>

</html>

