Name:Savaliya Heli Pareshbhai

RollNo: 071

Semester:7th

Subject:Application development using full stack

ASSIGNMENT:1

1. Develop a web server with following functionalities:

```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>
  <link rel="stylesheet" href="/styles.css">
</head>
```

```html
<body>
<form action="/process" method="get">
    Enter Name:<input type="text"
name="fname"><br><br>
    Eneter Age:<input
type="text"  name="age"><br><br>
    <input type="submit" value="submit">
</form>
</body>
```

```html
</html>
```

```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>
  <link rel="stylesheet" href="/styles.css">
</head>
```

```html
<body>
<form action="/process" method="POST">
    Enter Name:<input type="text"
name="fname"><br><br>
    Eneter Age:<input
type="text"  name="age"><br><br>
```

```
    <input type="submit" value="submit">
</form>
</body>
```

```
</html>
```

- Serve static resources.

```
const http = require('http');
const url = require('url');
const static = require('node-static');

const fileServer = new static.Server('./page');
```

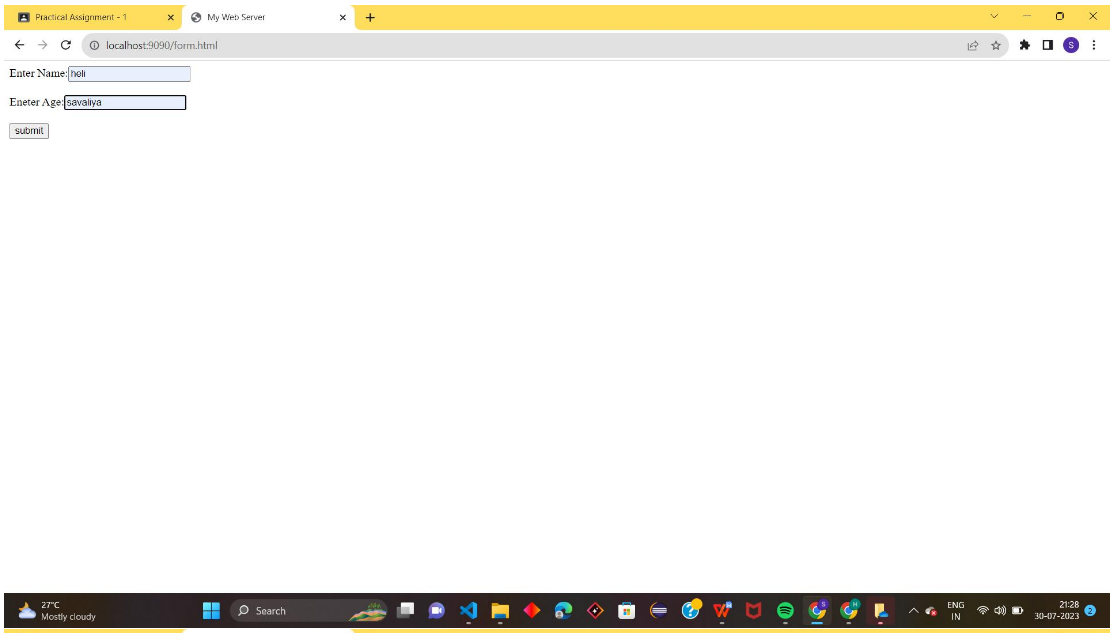
```
var server = http.createServer(function(request,
response) {
    //request.addListener('end',function(){}).resume()
;
    request.addListener('end', function () {
        fileServer.serve(request, response);
    }).resume();

}).listen(5000,()=>{
    console.log("Listening on port number 5000");
```

```
});
```

- Handle GET request.

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

const server = http.createServer((req, res) => {
    var u1 = url.parse(req.url,true);
//parseQueryString <boolean> If true, the query
property will always be set to an object returned by
the querystring module's parse() method. If false,
the query property on the returned URL object will be
an unparsed, undecoded string. Default: false.
```

```
    if (u1.pathname=="/process" && req.method ===
'GET')
    {
        res.write("Name: "+u1.query.fname+"
Age:  "+u1.query.age)
        res.end();
    }
    else if (req.url=="/form.html" && req.method ===
'GET')
    {
            var filename = "form.html";
            fs.readFile(filename, function(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);
            return res.end();
            });
    }
    else
    {
        res.write("/ is not allowed to access!!!");
        return res.end();
    }
});
server.listen(9090);
```

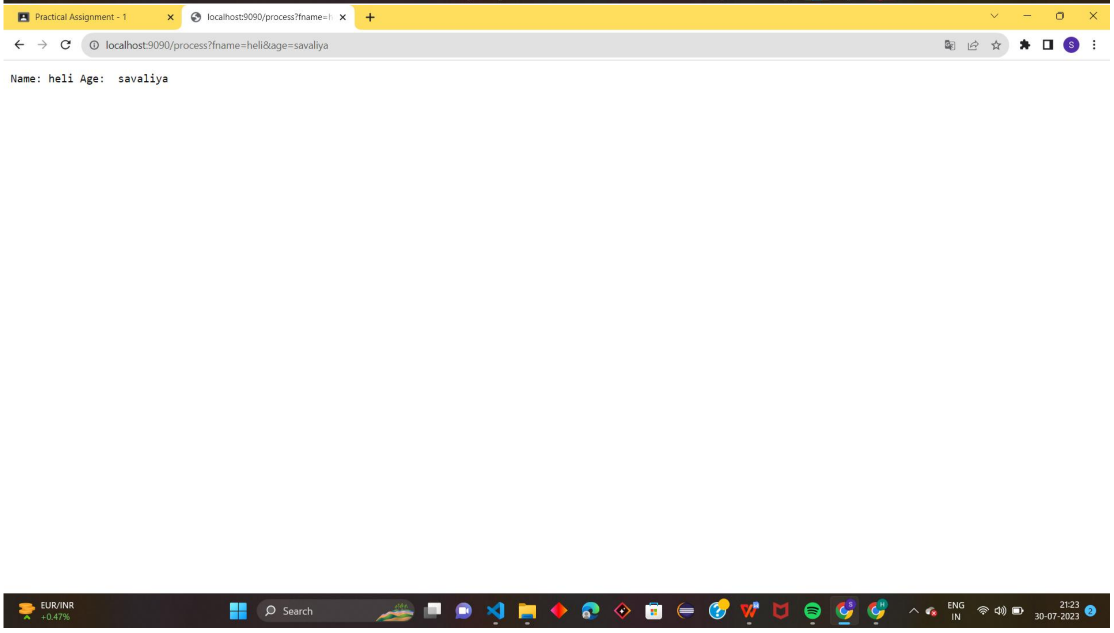


localhost:9090/form.html

Enter Name: heli

Eneter Age: savaliya

submit



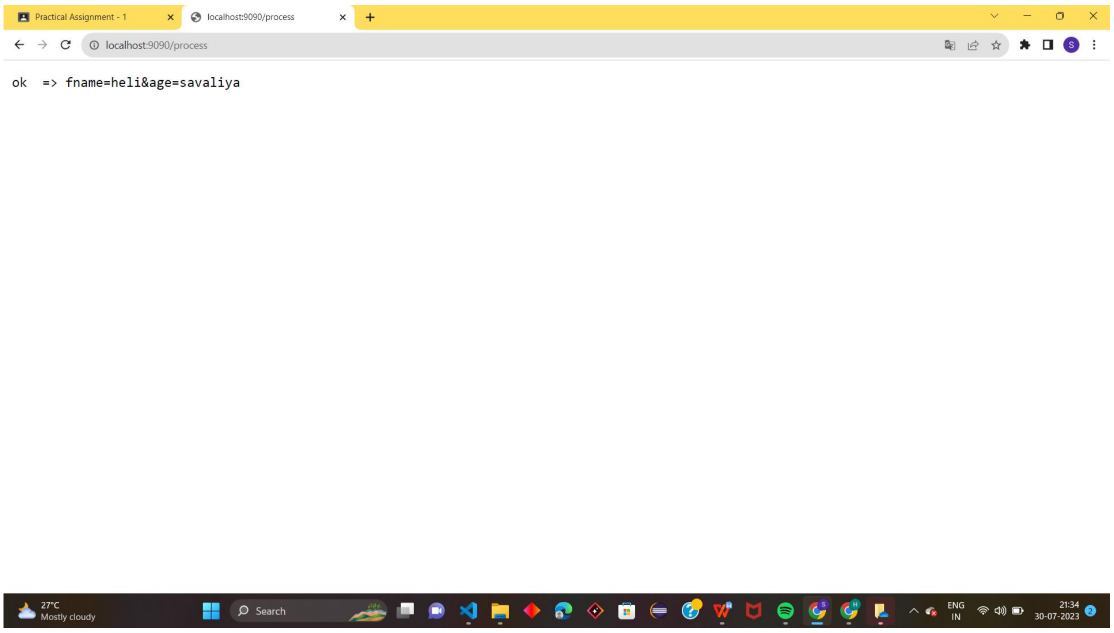
localhost:9090/process?fname=heli&age=savaliya

```
Name: heli Age:  savaliya
```

- Handle POST request.

```
const http = require('http');
const fs = require('fs');
//const server=http.createServer((req,res)=>{})
const server = http.createServer((req, res) => {
    if (req.url=="/process" && req.method === 'POST')
    {
        let body = '';
        //req.on("data",chunk=>{} )
        req.on('data', chunk => {
            body += chunk.toString(); // convert
Buffer to string
```

```
    });
    req.on('end', () => {
      console.log(body);
      res.end('ok  => '+ body);
      //fname=vibha&age=25
    });
  }
  else if (req.url=="/form_post.html" && req.method
=== 'GET')
  {
      var filename = "form_post.html";
      fs.readFile(filename, function(err, data)
{
      if (err) {
       res.writeHead(404, {'Content-Type':
'text/html'});
       return res.end("404 Not Found");
      }

      res.writeHead(200, {'Content-Type':
'text/html'});
      res.write(data);
      return res.end();
      });
  }
  else
  {
    res.write("/ is not allowed to access!!!");
    return res.end();
  }
});
server.listen(9090);

console.log("Server listening on port number 9090");
```



```
ok  => fname=heli&age=savaliya
```

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.)

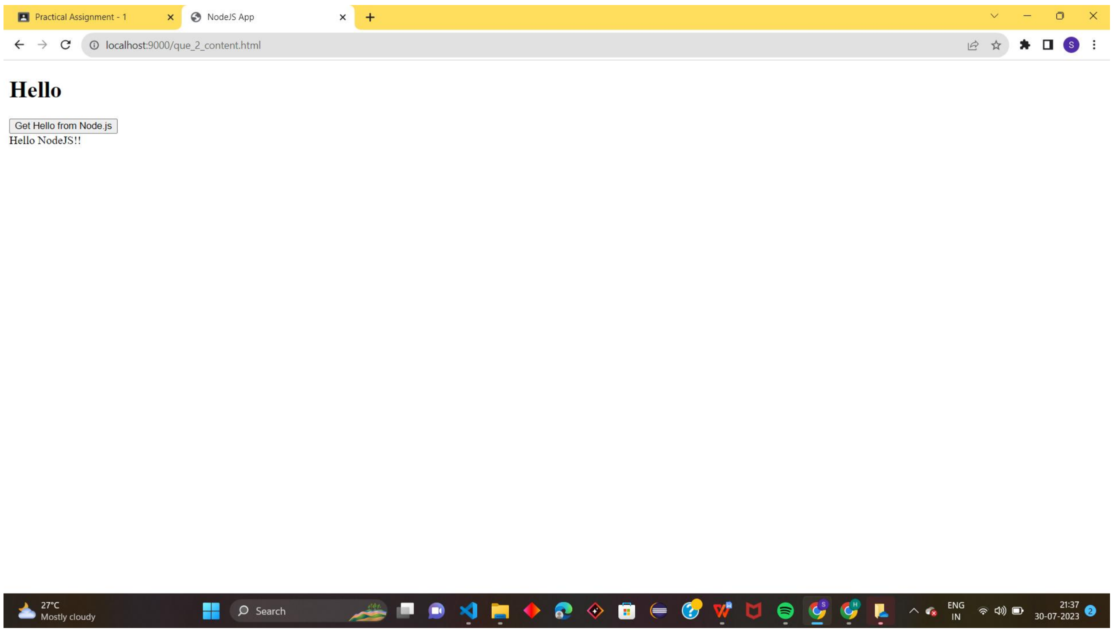
Html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>NodeJS App</title>
</head>
<body>
  <h1>Hello</h1>
  <button id="getHelloBtn">Get Hello from
Node.js</button>
  <div id="response"></div>

  <script>
    document.getElementById("getHelloBtn").addEventLi
stener("click", () => {
      fetch('/gethello')
        .then(response => response.text())
        .then(data => {
          document.getElementById("response").textCon
tent = data;
        })
        .catch(error => {
          document.getElementById("response").textCon
tent = "Error occurred: " + error.message;
        });
    });
  </script>
</body>
</html>
```

js

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

const server = http.createServer((req, res) => {
  if (req.url === "/gethello" && req.method === 'GET')
{
    res.writeHead(200, { 'Content-Type':
'text/plain' });
    res.end("Hello NodeJS!!");
  } else if (req.url === "/que_2_content.html" &&
req.method === 'GET') {
    // const filePath = path.join(__dirname,
"que_2_content.html");
    fs.readFile("que_2_content.html", (err, data) =>
{
      if (err) {
        res.writeHead(500, { 'Content-Type':
'text/plain' });
        res.end("Error reading the file");
      } else {
        res.writeHead(200, { 'Content-Type':
'text/html' });
        res.end(data);
      }
    });
  } else {
    res.writeHead(404, { 'Content-Type':
'text/plain' });
    res.end("404 Page1 Not Found");
  }
});
```

```
const port = 9000;
server.listen(port, () => {
  console.log(`Server listening on port ${port}`);
});
```



Hello

Hello NodeJS!!

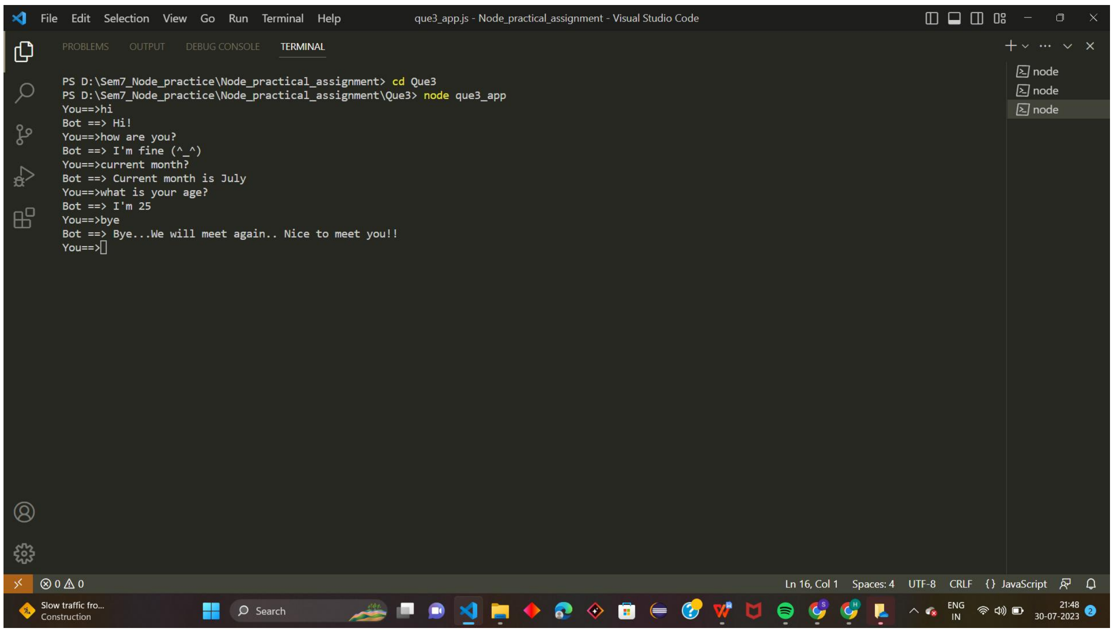
3.Develop a module for domain specific chatbot and use it in a command line application.
js

```
var Chatbot = require('./que3_chatbot');
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.ChatbotReply(message));
    //console.log('Bot ==>  '+ message);
    r1.prompt();
}).on('close',function(){  //chaining events.
    process.exit(0);
});
```

module

```
module.exports.ChatbotReply = function(message)
    {
        this.Bot_Age = 25;
        this.Bot_Name = "name1";
        this.Bot_University = "VNSGU";
        this.Bot_Country = "India";

        message= message.toLowerCase()

        if(message.indexOf("hi") > -1 ||
            message.indexOf("hello") > -1 ||
            message.indexOf("welcome") > -1 )
        {
            return "Hi!";
        }
        else if(message.indexOf("age") > -1 &&
            message.indexOf("your"))
        {
            return "I'm " + this.Bot_Age;
        }
        else if (message.indexOf("how") > -1 &&
            message.indexOf("are") &&
            message.indexOf("you"))
        {
            return "I'm fine (^_^)"
        }
        else if(message.indexOf("where") > -1
            && message.indexOf("live") &&
            message.indexOf("you"))
        {
            return "I live in " + this.Bot_Country;
        }
        else if(message.indexOf("bye") > -1 )
        {
            return "Bye...We will meet again.. Nice
to meet you!!"
        }
        return "Sorry, I didn't get it :( ";
    }
```



```
PS D:\Sem7_Node_practice\Node_practical_assignment> cd Que3
PS D:\Sem7_Node_practice\Node_practical_assignment\Que3> node que3_app
You==>hi
Bot ==> Hi!
You==>how are you?
Bot ==> I'm fine (^_^)
You==>current month?
Bot ==> Current month is July
You==>what is your age?
Bot ==> I'm 25
You==>bye
Bot ==> Bye...We will meet again.. Nice to meet you!!
You==>
```

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

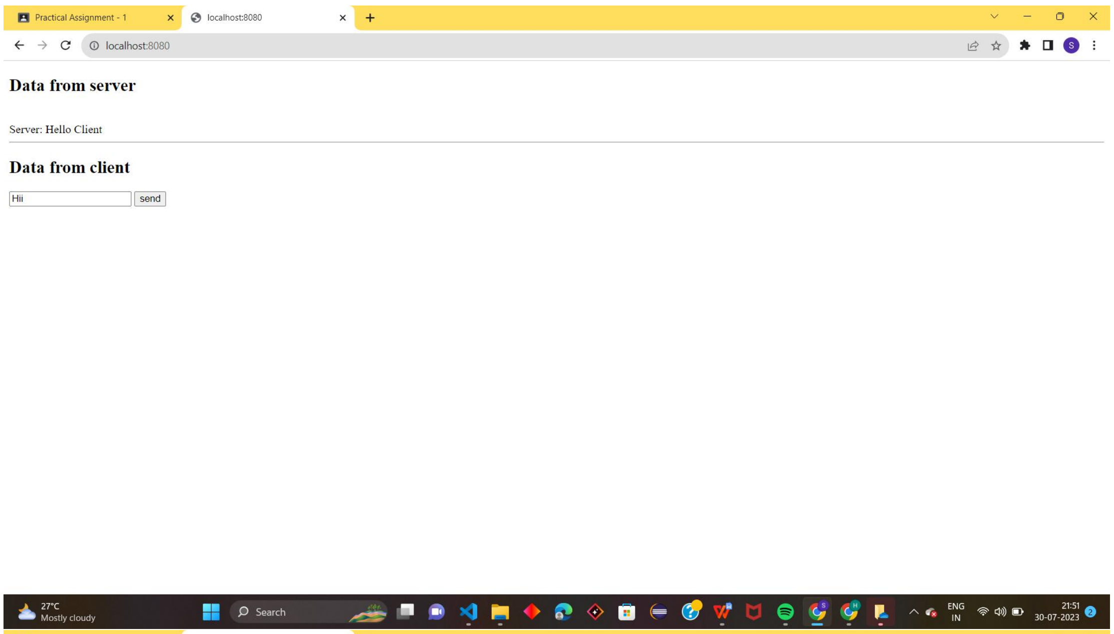
```
<!DOCTYPE html>
<html>

<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);
      document.getElementById('message').value = '';
    }
  </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>
```

JS

```
const WebSocket = require('ws')
var http = require('http');
var fs = require('fs');
const port = 8080;
var httpserver = http.createServer(function (request,
response) {
  if (request.url == "/") {
    fs.readFile("./public/index.html", (err, data) =>
{
      response.write(data)
      response.end();
    })
  }
}).listen({port}, function () {
  console.log((new Date()) +
    `Server is listening on
http://localhost:${port}`);
});
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)
  })
})
```



Practical Assignment - 1 | localhost:8080

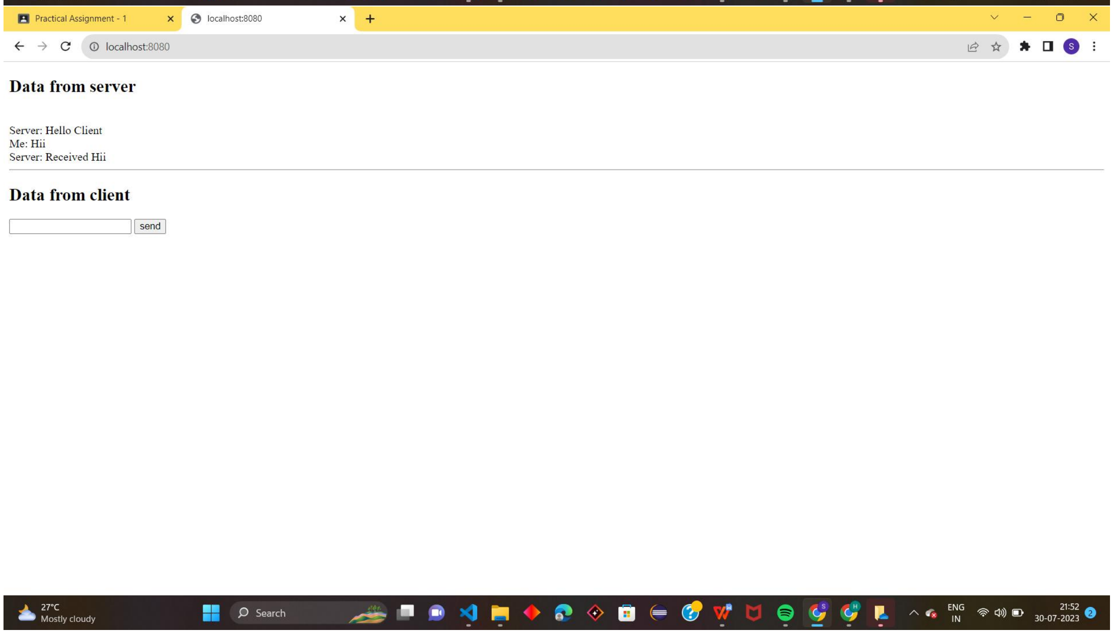
localhost:8080

Data from server

Server: Hello Client

Data from client

Hii | send



Practical Assignment - 1 | localhost:8080

localhost:8080

Data from server

Server: Hello Client
Me: Hii
Server: Received Hii

Data from client

send

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

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

function createZipArchive(sourceFolder, zipFileName)
{
  // Create a write stream for the zip file
  const output = fs.createWriteStream(zipFileName);
  // Create an archiver instance
  const archive = archiver('zip', {
    zlib: { level: 9 } // Compression level (0-9); 9
is the highest compression
  });
```

```
  // Listen for events to handle errors
  output.on('close', () => {
    console.log(`Archive created: ${zipFileName}`);
  });
```

```
  archive.on('warning', (err) => {
    if (err.code === 'ENOENT') {
      console.warn(err);
    } else {
      throw err;
    }
  });
```

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

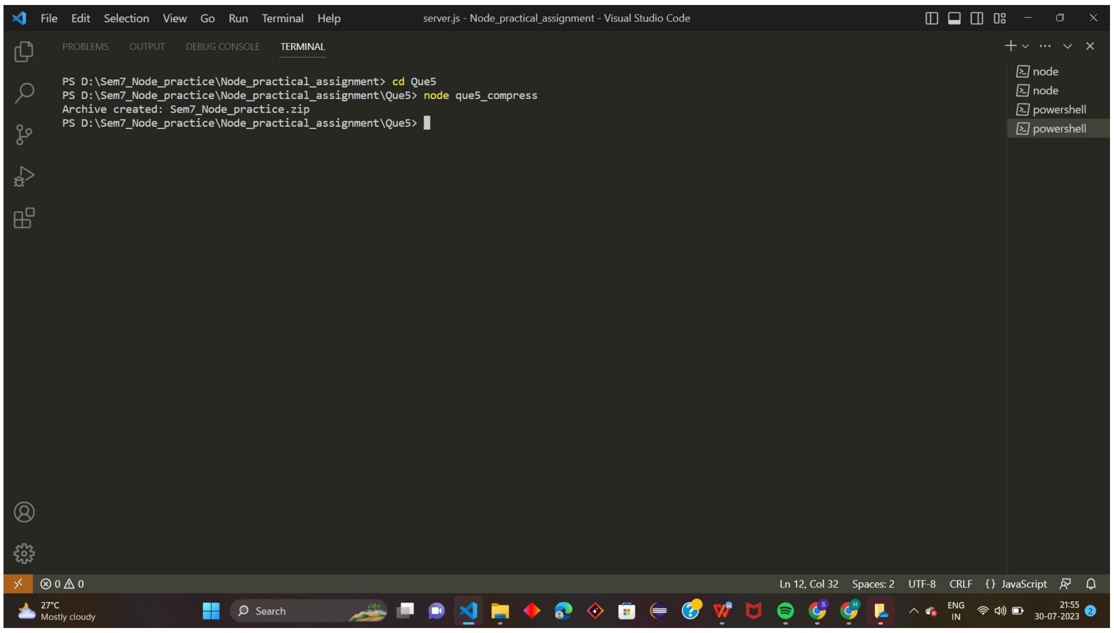
```
  // Pipe the archive data to the output stream
  archive.pipe(output);
```

```
  // Append the folder to the archive
  archive.directory(sourceFolder, false);
```

```
  // Finalize the archive (writes the zip file)
  archive.finalize();
}
```

```
// Usage example:
const sourceFolder = './Sem7_Node_practice'; //
Replace with the path to your folder
const zipFileName = 'Sem7_Node_practice.zip'; //
Replace with the desired zip file name
```

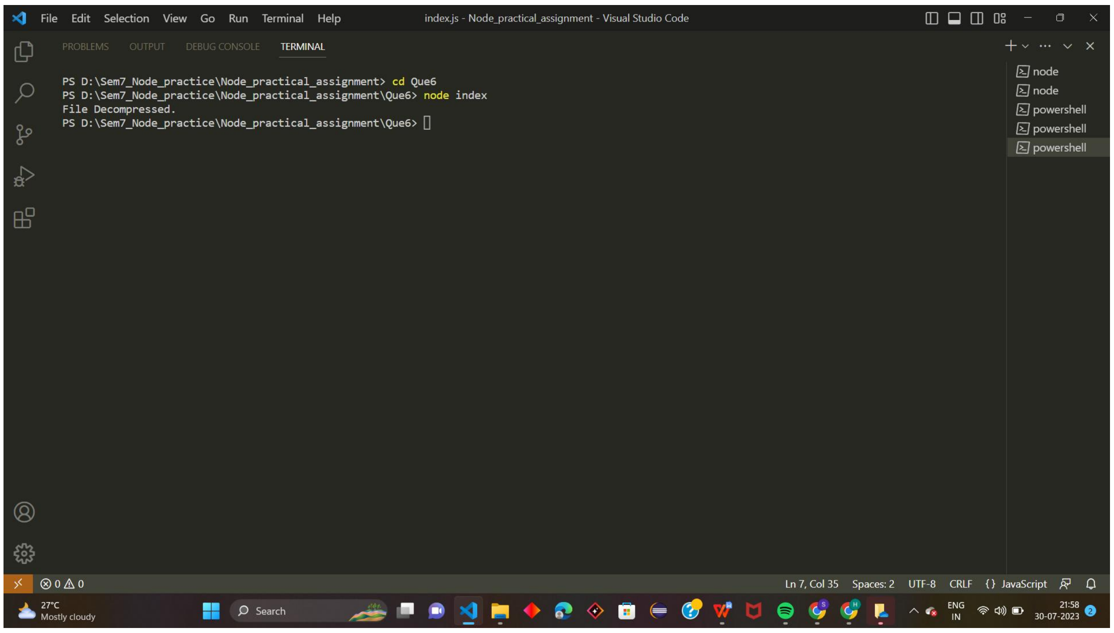
```
createZipArchive(sourceFolder, zipFileName);
```



```
PS D:\Sem7_Node_practice\Node_practical_assignment> cd Que5
PS D:\Sem7_Node_practice\Node_practical_assignment\Que5> node que5_compress
Archive created: Sem7_Node_practice.zip
PS D:\Sem7_Node_practice\Node_practical_assignment\Que5>
```

6. Write a program to extract a zip file.

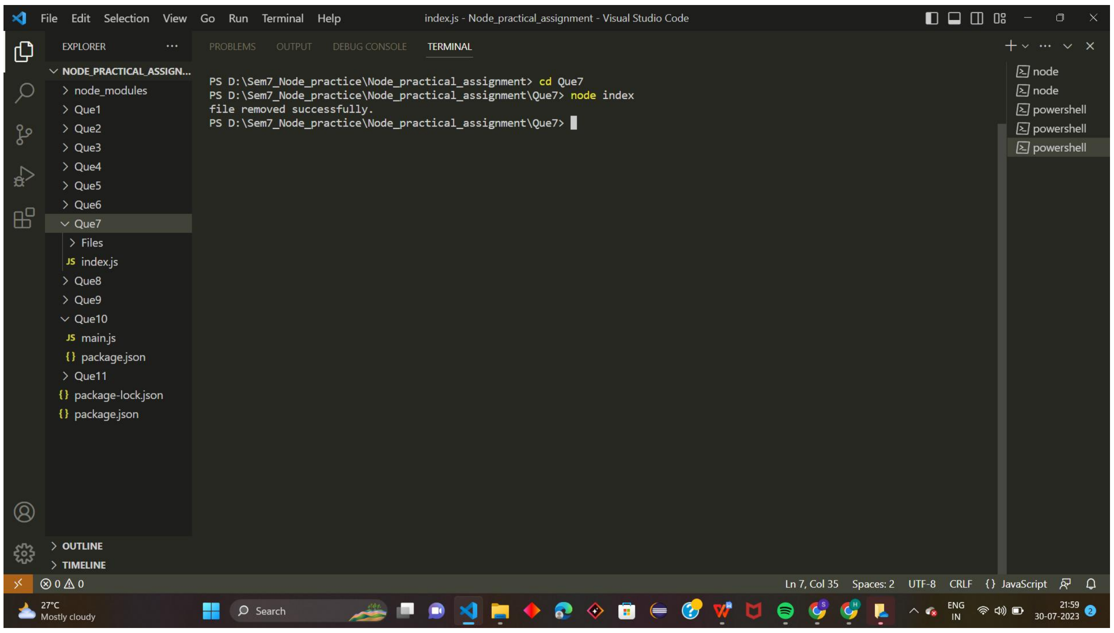
```
var fs = require("fs");
var zlib = require('zlib');

fs.createReadStream('./Files/text1.txt.gz')
  .pipe(zlib.createGunzip())
  .pipe(fs.createWriteStream('./Files/text1.txt',
'utf-8'));
console.log("File Decompressed.");
```



```
PS D:\Sem7_Node_practice\Node_practical_assignment> cd Que6
PS D:\Sem7_Node_practice\Node_practical_assignment\Que6> node index
File Decompressed.
PS D:\Sem7_Node_practice\Node_practical_assignment\Que6>
```

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

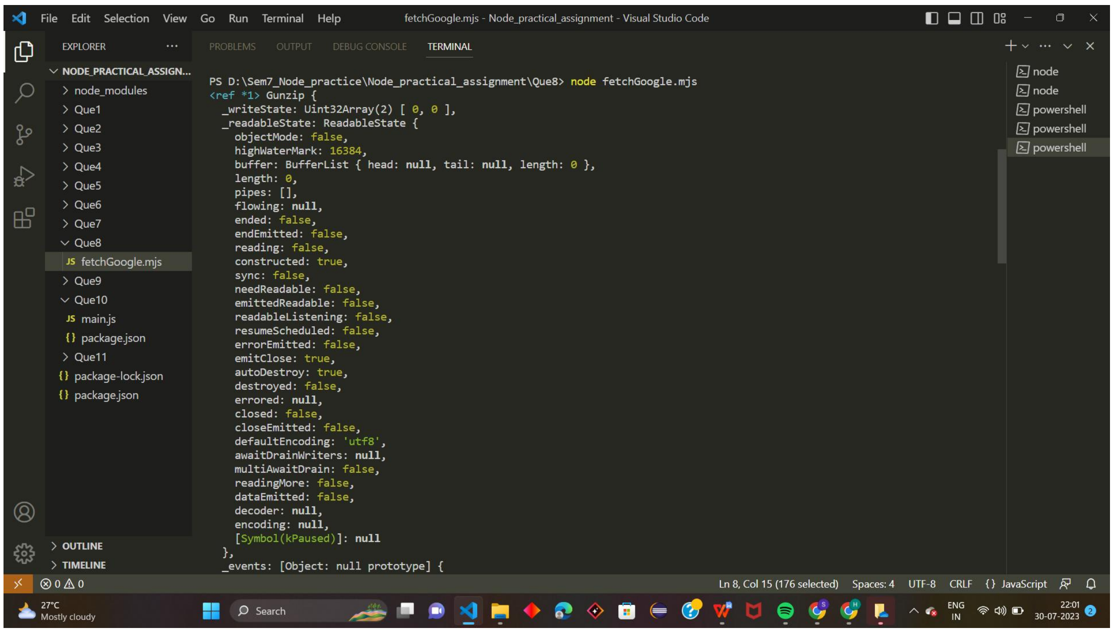
```
const fs = require("fs")
const removeFile = (file_path) => {
  return new Promise((resolve, reject) => {
    fs.unlink(file_path, (err) => {
      if (err) {
        return reject(err)
      }
      else {
        return resolve('file removed successfully.')
      }
    })
  })
}
removeFile('./files/text1.txt').then(msg => {
  console.log(msg)
}).catch(error => {
  console.log('error occured while deleting file ' + 
error)
})
```



```
PS D:\Sem7_Node_practice\Node_practical_assignment> cd Que7
PS D:\Sem7_Node_practice\Node_practical_assignment\Que7> node index
file removed successfully.
PS D:\Sem7_Node_practice\Node_practical_assignment\Que7> 
```

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

```
import fetch from 'node-fetch';

async function fetchGoogle() {
    const result = await
fetch('https://www.google.com/')
    console.log(result.body);
}
fetchGoogle();
```



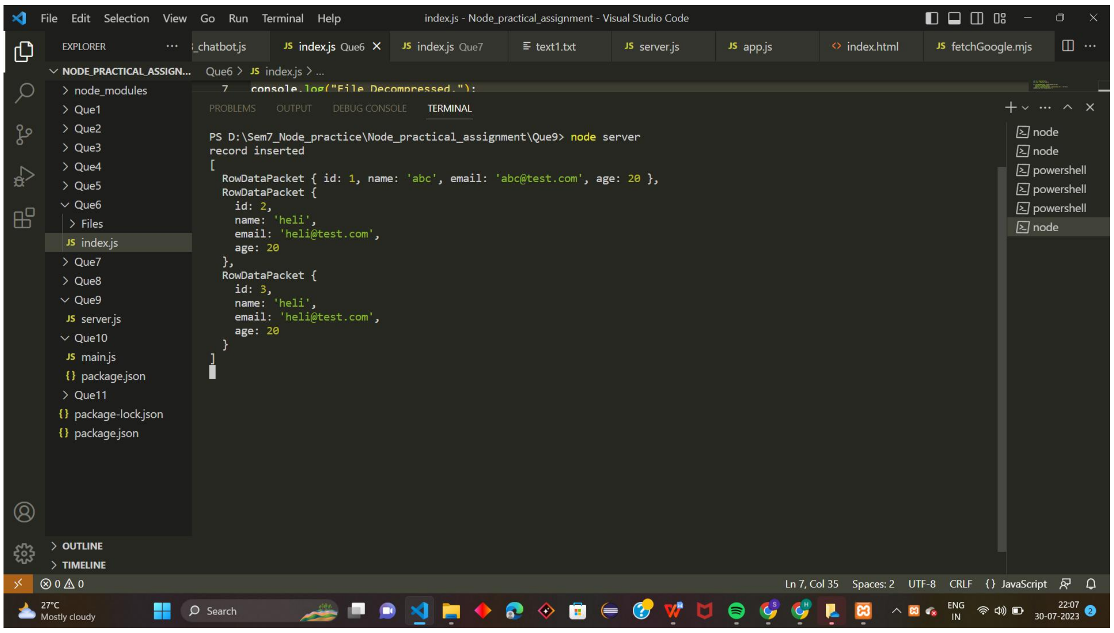
```
PS D:\Sem7_Node_practice\Node_practical_assignment\Que8> node fetchGoogle.mjs
<ref *1> Gunzip {
  _writeState: Uint32Array(2) [ 0, 0 ],
  _readableState: ReadableState {
    objectMode: false,
    highWaterMark: 16384,
    buffer: BufferList { head: null, tail: null, length: 0 },
    length: 0,
    pipes: [],
    flowing: null,
    ended: false,
    endEmitted: false,
    reading: false,
    constructed: true,
    sync: false,
    needReadable: false,
    emittedReadable: false,
    readableListening: false,
    resumeScheduled: false,
    errorEmitted: false,
    emitClose: true,
    autoDestroy: true,
    destroyed: false,
    errored: null,
    closed: false,
    closeEmitted: false,
    defaultEncoding: 'utf8',
    awaitDrainWriters: null,
    multiAwaitDrain: false,
    readingMore: false,
    dataEmitted: false,
    decoder: null,
    encoding: null,
    [Symbol(kPaused)]: null
  },
  _events: [Object: null prototype] {
```

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.

```
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 {
```

```
    con.query("INSERT INTO empTB
values(null,'heli','heli@test.com',20)", (err, result)
=> {
```

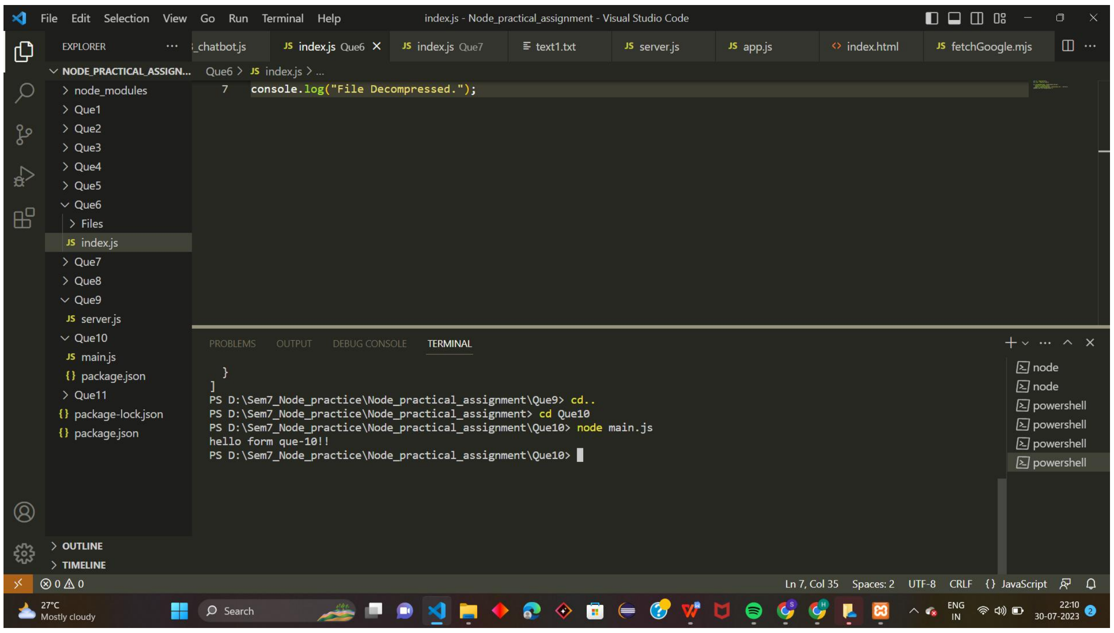
```
if (err) {
      console.log("error: " + err)
    } else {
      console.log("record inserted")
    }
  })
selectAllEmployees().then(result => {
  console.log(result)
}).catch(err => {
  console.log("error: " + err)
})
}
})
```



```
PS D:\Sem7_Node_practice\Node_practical_assignment\Que9> node server
record inserted
[
  RowDataPacket { id: 1, name: 'abc', email: 'abc@test.com', age: 20 },
  RowDataPacket {
    id: 2,
    name: 'heli',
    email: 'heli@test.com',
    age: 20
  },
  RowDataPacket {
    id: 3,
    name: 'heli',
    email: 'heli@test.com',
    age: 20
  }
]
```

10. Set a server script, a test script and 3 user defined scripts in package.json file in your nodejs
application.
Main.js
```
console.log("hello form que-10!!");
```

json
```
{
  "name": "q-10",
  "version": "1.0.0",
  "description": "",
  "main": "main.js",
  "scripts": {
    "test": "test.js",
    "start": "main.js",
    "myscript": "main.js",
    "script1": "main.js",
    "script2": "main.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "node-static": "^0.7.11"
  }
}
```



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

```
  }
]
PS D:\Sem7_Node_practice\Node_practical_assignment\Que9> cd..
PS D:\Sem7_Node_practice\Node_practical_assignment> cd Que10
PS D:\Sem7_Node_practice\Node_practical_assignment\Que10> node main.js
hello form que-10!!
PS D:\Sem7_Node_practice\Node_practical_assignment\Que10> 
```