

Name : Varun R. Rajput

Roll no : 56

Msc ICT SEM-3

Subject – Open source web development

Q1. Develop a web server with following functionalities:

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

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Home - My Website</title>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <nav>
      <ul>
        <li><a href="/index.html">Home</a></li>
        <li><a href="/page1.html">Page 1</a></li>
        <li><a href="/page2.html">Page 2</a></li>
      </ul>
    </nav>
```

```
<header>
```

```
<h1>Welcome to My Website</h1>
```

```
<p>Explore the content through the navigation above.</p>
```

```
</header>
```

```
<section class="main-content">
```

```
<h2>About This Site</h2>
```

```
<p>
```

```
    This is a simple website with multiple pages to demonstrate HTML and  
    CSS.
```

```
</p>
```

```
</section>
```

```
<footer>
```

```
<p>&copy; 2024 My Website. All rights reserved.</p>
```

```
</footer>
```

```
</body>
```

```
</html>
```

Page2.html

```
<!DOCTYPE html>
```

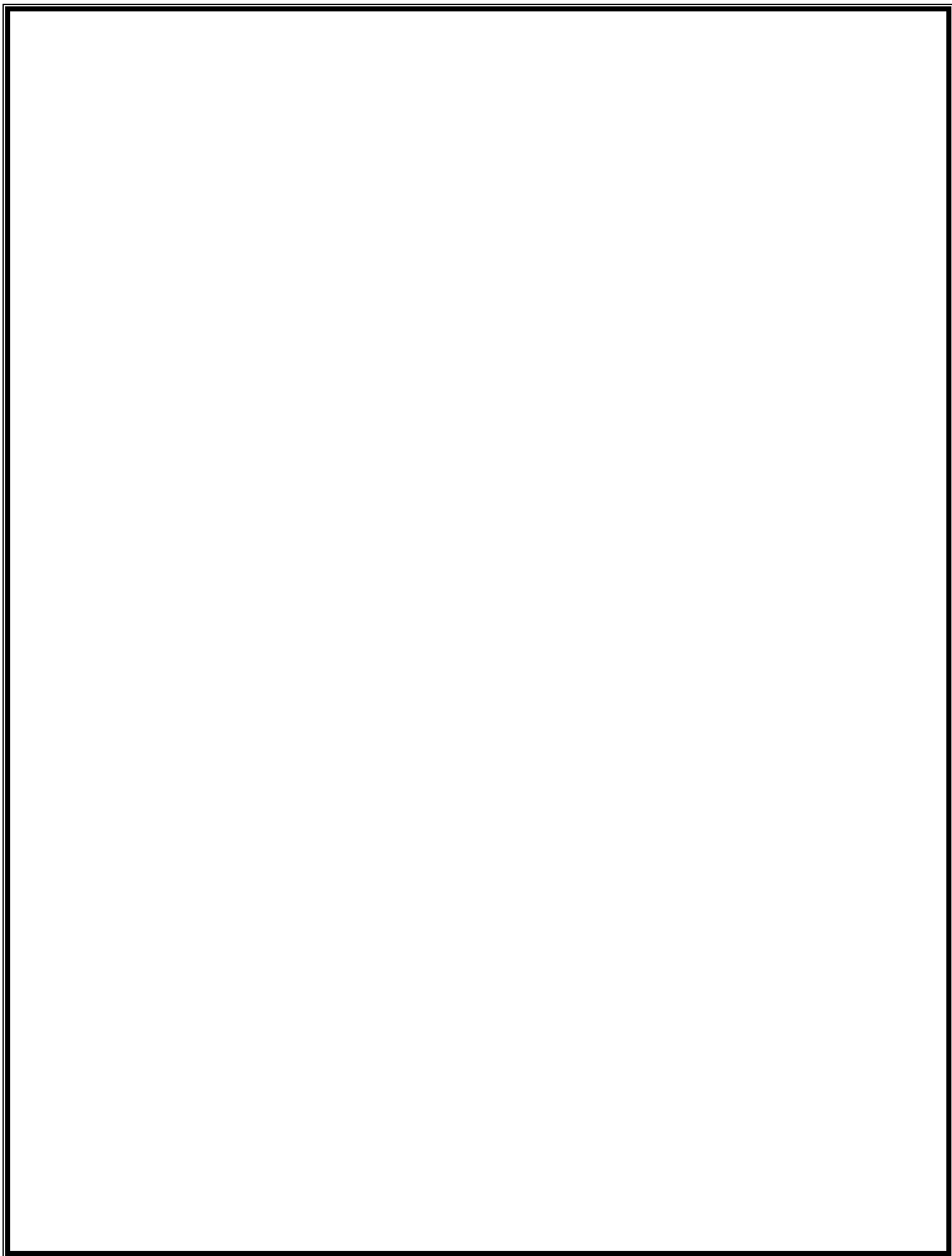
```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8" />
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Page 2</title>
<link rel="stylesheet" href="./style.css" />
</head>
<body>
  <nav>
    <ul>
      <li><a href="./index.html">Home</a></li>
      <li><a href="./page1.html">Page 1</a></li>
      <li><a href="./page2.html">Page 2</a></li>
    </ul>
  </nav>

  <div class="formDiv" >
    <form method="post" action="handleSubmit">
      <input type="text" name="name" placeholder="Enter name" />
      <input type="email" name="email" placeholder="Enter Email" />
      <input type="tel" name="phone" placeholder="Enter Phone" />
      <button type="submit">Submit</button>
    </form>
  </div>
</body>
</html>
```



Index.js

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

```
const port = 8000;
```

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

```
app.use(express.static("public"));
```

```
app.use(express.urlencoded({ extended: true }));
```

```
app.post("/handleSubmit", (req, res) => {
```

```
  const { name, email, phone } = req.body;
```

```
  res.send(`Data submitted ${name}`);
```

```
  if (name !== null) {
```

```
    console.log(name);
```

```
  }
```

```
});
```

```
app.listen(port, () => {
```

```
  console.log(`Your app is listing at : http://localhost:${port}`);
```

```
});
```

← → ↻ ⓘ localhost:8000/handleSubmit

Data submitted varun

Q2. 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.)

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>
  </head>
  <body>
    <h1>Node.js AJAX Example</h1>
    <button id="getHelloBtn">Get Hello Message</button>
    <p id="message"></p>

    <script>
      document
        .getElementById("getHelloBtn")
```



```
.addEventListener("click", function () {  
  fetch("/gethello")  
    .then((response) => response.text())  
    .then((data) => {  
      document.getElementById("message").innerText = data;  
    })  
    .catch((error) => console.error("Error:", error));  
});  
</script>  
</body>  
</html>
```

Server.js

```
const express = require("express");  
const path = require("path");  
const app = express();  
  
// Serve the HTML file  
app.get("/", (req, res) => {  
  res.sendFile(path.join(__dirname, "index.html"));  
});
```

```
app.get("/gethello", (req, res) => {  
  res.send("Hello NodeJS!!");  
});  
  
const PORT = process.env.PORT || 3000;  
app.listen(PORT, () => {  
  console.log(`Server is running on port ${PORT}`);  
});
```

Node.js AJAX Example

Get Hello Message

Hello NodeJS!!

Q3. Develop a module for domain specific chatbot and use it in a command line application.

```
// chatbot.js
```

```
class Chatbot {  
  constructor(domain) {  
    this.domain = domain;  
    this.responses = {  
      greeting: `Hello! How can I assist you with ${this.domain}?`,  
      help: `I can help you with queries related to ${this.domain}. What would you like to know?`,  
      goodbye: "Goodbye! Have a great day!",  
      fallback:  
        "I'm not sure how to respond to that. Can you please ask something else?",  
    };  
  }  
  
  getResponse(input) {  
    const normalizedInput = input.toLowerCase();  
  
    if (normalizedInput.includes("hello") || normalizedInput.includes("hi")) {  
      return this.responses.greeting;  
    } else if (normalizedInput.includes("help")) {  
      return this.responses.help;  
    }  
  }  
}
```

```
    } else if (normalizedInput.includes("bye")) {  
        return this.responses.goodbye;  
    } else {  
        return this.responses.fallback;  
    }  
}  
}
```

```
module.exports = Chatbot;
```

index.js

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

```
const domain = "web development"; // You can change the domain as needed  
const bot = new Chatbot(domain);
```

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

```
const askQuestion = () => {
```

```
rl.question("You: ", (input) => {  
  const response = bot.getResponse(input);  
  console.log(`Bot: ${response}`);  
  
  if (input.toLowerCase().includes("bye")) {  
    rl.close();  
  } else {  
    askQuestion();  
  }  
});  
};  
  
console.log(`Welcome to the ${domain} chatbot!`);  
askQuestion();
```

```
PS E:\Node-Assignment\q3> node index  
Welcome to the web development chatbot!  
You: hello  
Bot: Hello! How can I assist you with web development?  
You: help  
Bot: I can help you with queries related to web development. What would you like to know?  
You: bye  
Bot: Goodbye! Have a great day!  
PS E:\Node-Assignment\q3> █
```

Q5. Write a program to create a compressed zip file for a folder.

Index.js

```
const express = require("express");
const fileUpload = require("express-fileupload");
const archiver = require("archiver");
const fs = require("fs-extra");
const path = require("path");

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

// Middleware to handle file uploads
app.use(fileUpload());

// Serve static files from the 'public' directory
app.use(express.static(path.join(__dirname, "public")));

// Route to handle folder compression
app.post("/compress", async (req, res) => {
  if (!req.files || Object.keys(req.files).length === 0) {
    return res.status(400).send("No files were uploaded.");
  }
});
```

```
const files = req.files.files; // Array of files
const outputFilePath = path.join(__dirname, "compressed", "output.zip");

// Ensure the compressed directory exists
await fs.ensureDir(path.join(__dirname, "compressed"));

// Create the zip file
const output = fs.createWriteStream(outputFilePath);
const archive = archiver("zip", { zlib: { level: 9 } }); // Compression level: 0-9

output.on("close", () => {
  console.log(`${archive.pointer()} total bytes`);
  console.log("Zip file has been created successfully.");
  res.download(outputFilePath);
});

archive.on("error", (err) => {
  res.status(500).send(`Error creating zip file: ${err.message}`);
});

archive.pipe(output);

// Add each file to the zip
if (Array.isArray(files)) {
```

```
files.forEach((file) => {  
  archive.append(file.data, { name: file.name });  
});  
} else {  
  archive.append(files.data, { name: files.name });  
}  
  
await archive.finalize();  
});  
  
// Start the server  
app.listen(PORT, () => {  
  console.log(`Server is running 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>Compress Folder to Zip</title>  
  </head>  
  <body>
```



```
<h1>Compress Folder to Zip</h1>
<form action="/compress" method="post" enctype="multipart/form-data">
  <label for="folder">Select files from a folder:</label>
  <input
    type="file"
    name="files"
    id="files"
    webkitdirectory
    directory
    multiple
    required
  />
  <button type="submit">Compress and Download</button>
</form>
</body>
</html>
```

Compress Folder to Zip

Select files from a folder:

Choose Files

No file chosen

Compress and Download

Compress Folder to Zip

Select files from a folder:

Choose Files

No file chosen

Compress and Download

Upload 2 files to this site?

This will upload all files from "question1". Only do this if you trust the site.

Upload

Cancel

Compress Folder to Zip

Select files from a folder:

Choose Files

2 files

Compress and Download

📁

output.zip

1.794 B • Done

Q6. Write a program to extract a zip file.

Index.js

```
const express = require("express");
const fileUpload = require("express-fileupload");
const unzipper = require("unzipper");
const fs = require("fs");
const path = require("path");
const { Readable } = require("stream");

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

// Middleware to handle file uploads
app.use(fileUpload());

// Serve static files from the 'public' directory
app.use(express.static(path.join(__dirname, "public")));

// Function to convert buffer to stream
function bufferToStream(buffer) {
  const stream = new Readable();
  stream.push(buffer);
  stream.push(null); // Indicates the end of the stream
}
```

```
    return stream;
  }
  app.post("/upload", (req, res) => {
    if (!req.files || !req.files.zipfile) {
      return res.status(400).send("No file was uploaded.");
    }

    const zipFile = req.files.zipfile;
    const extractPath = path.join(__dirname, "extracted");

    if (!fs.existsSync(extractPath)) {
      fs.mkdirSync(extractPath);
    }

    const zipStream = bufferToStream(zipFile.data);
    zipStream
      .pipe(unzipper.Extract({ path: extractPath }))
      .on("close", () => {
        res.send("File extracted successfully!");
      })
      .on("error", (err) => {
        res.status(500).send(`Error extracting file: ${err.message}`);
      });
  });
```

```
app.listen(PORT, () => {  
  console.log(`Server is running 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>Upload and Extract ZIP File</title>  
  </head>  
  <body>  
    <h1>Upload a ZIP File</h1>  
    <form action="/upload" method="post" enctype="multipart/form-data">  
      <input type="file" name="zipfile" accept=".zip" required />  
      <button type="submit">Upload and Extract</button>  
    </form>  
  </body>  
</html>
```

Upload a ZIP File

Choose File

LoginRegister.zip

Upload and Extract



localhost:3000/upload

File extracted successfully!

Q7. Write a program to promisify fs.unlink function and call it.

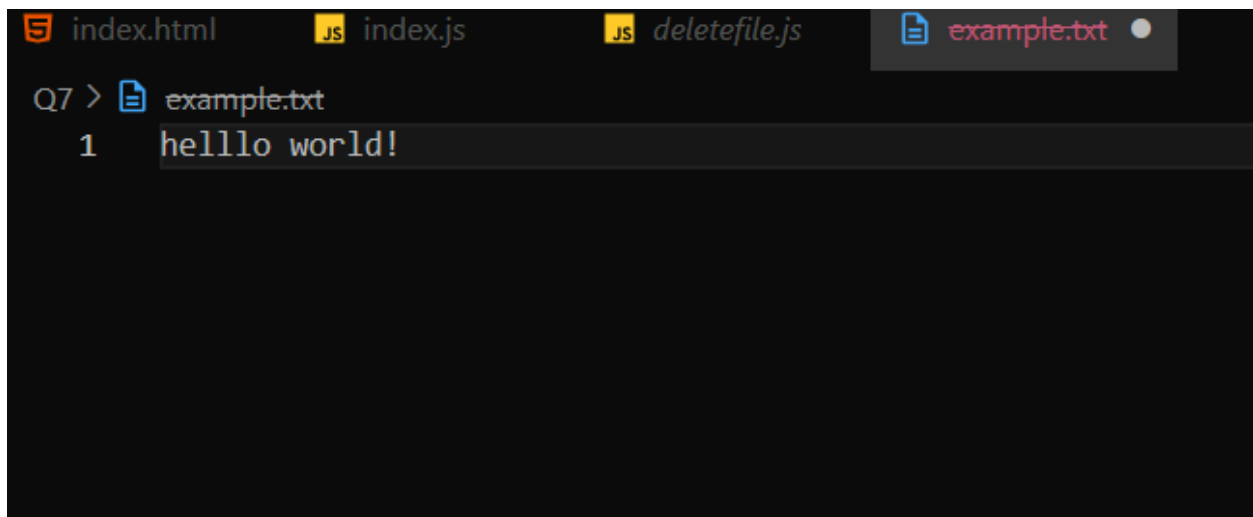
Deletefile.js

```
const fs = require("fs");
const util = require("util");
const unlinkAsync = util.promisify(fs.unlink);
async function deleteFile(filePath) {
  try {
    await unlinkAsync(filePath);
    console.log(`File ${filePath} was deleted successfully.`);
  } catch (error) {
    console.error(`Error deleting file ${filePath}:`, error);
  }
}
const filePath = "example.txt";
deleteFile(filePath);
```

example.txt

hello world!!!!

```
PS E:\Node-Assignment> cd q7
PS E:\Node-Assignment\q7> node deletefile
File example.txt was deleted successfully.
PS E:\Node-Assignment\q7>
```



The screenshot shows a code editor interface with four tabs at the top: 'index.html' (with an HTML icon), 'index.js' (with a JS icon), 'deletefile.js' (with a JS icon), and 'example.txt' (with a text icon and a close button). The 'example.txt' tab is active. Below the tabs, the editor shows the file path 'Q7 > example.txt' and a single line of code: '1 hello world!'.

Q8. Fetch data of google page using node-fetch using async-await model.

Index.js

```
async function fetchGoogleHomePage() {  
  try {  
    const fetch = (await import("node-fetch")).default;  
  
    const response = await fetch("https://www.google.com");  
  
    if (!response.ok) {  
      throw new Error(`HTTP error! status: ${response.status}`);  
    }  
  
    const data = await response.text();  
  
    console.log(data);  
  } catch (error) {  
    console.error("Error fetching Google homepage:", error);  
  }  
}  
  
fetchGoogleHomePage();
```

```
PS E:\assignnode\NodeJS Assignment\Question8> node index
<!doctype html><html ixscope="" itemtype="http://schema.org/WebPage" lang="en-IN"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/google1x/google_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="voCwKqRQ9z2809oN6RSUg">(function(){var g={KEI:"Kga6ZyQFNtcl0p9eqauqs","XEPI":0,2504525,1195730,687,439,5,92,538567,2882,2891,11754,31274,30022,16105,18161,60057,85641,2,16737,23024,6700,41942,57740,2,2,1,23825,10962,8861,14489,8703,1733,9779,62657,33566,39613,3030,15816,1804,7734,27535,2413,9877,1158,9707,19569,11106,3081,12896,5203197,6231,3248,977,21,64,63,5991305,846,2839947,29,2,355,1,1,32,26816545,164377,16672,43887,3,318,4,1281,3,2124363,23029351,7954,1,208,10336,10736,34760,5,1899,24916,22465,11643,86,10893,12953,2212,7146,1035,154,10653,9908,26880,1835,9236,12431,675,155,2,2482,13503,7737,6597,2,2539,1995,2605,168,6,154,1543,1674,4,3004,6754,832,5,12083,409,93,425,3764,1955,1670,5633,687,6592,85,1119,3,5,413,1353,1244,2695,2757,249,1469,234,377,476,216,5568,687,1315,5402,440,339,6372,1150,49,1440,437,286,2,9,3212,2,1,782,3,5,1209,6,3,3343,295,405,8,2359,1775,2379,567,917,799,182,306,222,1858,2,907,3792,1,194,2955,2023,6,698,1367,1788,142,3,2,3270,446,1252,7088,6,1156,462,913,1050,703,49,45,2236,156,152,64,2,28,880,1556,767,1778,37,562,4,455,1,1898,185,178,32,3,352,189,384,13659,441,111,4,507,106,61,29,190,55,123,602,3,447,2,4,972,387,732,231,1144,353,7750,660,369,160,1184,370,273,213,369,246,906,2,5,2,304,270,287,2,3,493,7,1049,61,161,918,379,2608,13,134,140,160,3,25,206,281,50,131,374,299,3,93,472,116,203,32,12,603,3,736,739,3,180,136,2,587,508,2,3,494,1304,413,402,180,70,267,169,137,40,426,102,438,719,175,62,13,234,39,4,1245,3,154,606,575,266,321,1,8,1,1,4,107,659,3,103,1034,805,915,681,3,260,71,811,458,267,21502732,3,14475,3,2919,4134,1112,47,608',"kBL":"l4v7","KOPI:89978449");(function(){var a;((a===window.google)==null?0:a.stvs)<google.kEI=g.kEI;window.google.g=g;}.call(this));})(function(){google.sn=webhp;google.khl='en-IN');})(function(){var h=this||self;function l(){return window.google!==void 0&&window.google.KOPI!==(void 0&&window.google.KOPI==0?window.google.KOPI:null);var m,n;function p(a){for(var b;a=&(!a.getAttribute)||!(b=a.getAttribute("eid")));a=a.parentNode;return b|n}function q(a){for(var b=null;&&(!a.getAttribute)||!(b=a.getAttribute("leid")));a=a.parentNode;return b}function r(a,b)/<http://i.test(a)&&window.location.protocol=="https"&&(google.ml&&google.ml(Error(a)),1,{src:a,gLmm:1}),a====);return a}function t(a,b,c,d,k){var e====;"b.search("&ei="+&e+"p(d),b=search("&lei="+&e+"&d(q(d))&&e+("&ei="+&e+"&"+&e+"d)";var g=b.search("&cschid")====&lei==""slh",f=[];f.psh("&xz",Date.now(),t.toString());h.cshid&&g&&.push(["cshid",h.cshid]);c=c|null&&.push(["opi",c.toString()]);for(c<0;c<f.length;++){if(c===0){c0=d="8";d=f[c][0]+&f[c][1]}return"/"+k["gen_204"]+"&atyp=i&ct="+String(a)+"&cad="+&b+(e+d)};m=google.kEI;google.getEIQ=p;google.getLEIQ=g;google.lm=function(){return null};google.log=function(b,c,d,k,e){e===void 0?l:e||(c=t(c,t(a,b,e,d,k)));if(c=(c))&a=new Image;var g=n.length;n[g]=a.a.onerror=a.onload=a.onabort=function(){delete n[g];a.src=c}};google.logUrl=function(a,b){b===void 0?l:b}<function t("",a,b)}).call(this);(function(){google.y={};google.sy={};var d;(d=google).x|[(d.x=function(a,b){if(a)var c=a.id}else{do c=Math.random()while(google.y[c])};google.y[c]=[a,b];return l});var e;(e=google).sx|(e.sx=function(a){google.sy.push(a)});google.lm={};var f;(f=google).plm|(f.plm=function(a){google.lm.push.apply(google.lm,a)});google.lq=[f];var g;(g=google).load|(g.load=function(a,b,c){google.lq.push([f[a],b,c]));var h;(h=google).loadAll|(h.loadAll=function(a,b){google.lq.push([f[a],b]));google.bx=!var k;(k=google).lx|(k.lx=function()){var l=[f,m;(m=google).fce|(m.fce=function(a,b,c,n){l.push([a,b,c,n]));google.qce=l}).call(this);google.f=(function(){document.documentElement.addEventListener("submit"),function(b){var a;if(a=b.target).var c=a.getAttribute("data-submitfa");a===""?"&l|=a.elements.value?0:l}|els a=!1;&&b.preventDefault(),b.stopPropagation()),l0);document.documentElement.addEventListener("click",function(b){var a;{for(a=b.target;&&a!=document.documentElement;a.parentNode)if(a.tagName==="A"){a=a.getAttribute("data-noref")===""?1;break;a=!1}&&b.preventDefault(),l0}}).call(this);<script><style>#gbar,#guser{font-size:13px;padding-top:1px !important;#gbar{height:22px}#guser{padding-bottom:7px !important;text-align:right}.gbh,.gbd{border-top:1px solid #c9d7f1;font-size:1px}.gbh{height:0;position:absolute;top:24px;width:100%}@media all{.gb1{height:22px;margin-right:.5em;vertical-align:top}#gbar{float:left}.gb1,a.gb4{text-decoration:underline !important}.gb1,a.gb4{color:#00c !important}.gb1 .gb4{color:#dd8e27 !important}.gbf .gb4{color:#900 !important}</style><style>body,td,.gh,.hfont-family:arial,sans-serif;font-size:100%;overflow-y:scroll}#ggog(padding:3px 8px 0)td{line-height:.8em}.gam_m td{line-height:17px}form{margin-bottom:20px}.h{color:#a967d2}em{font-weight:bold;font-style:normal}.lst{height:25px;width:496px}.gsfi,.lst{font:18px arial,sans-serif}.gsfs{font:17px arial,sans-serif}.ds{disp
```

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("mysql2/promise");
```

```
// Database configuration
```

```
const config = {
```

```
  host: "localhost",
```

```
  user: "root",
```

```
  password: "varun@123",
```

```
  database: "employee",
```

```
};
```

```
async function connectDatabase() {
```

```
  try {
```

```
    const connection = await mysql.createConnection(config);
```

```
    console.log("Connected to the database.");
```

```
// Insert a record into the employee table
```

```
const insertQuery = `INSERT INTO empdata(name,department,age) VALUES (?, ?, ?)`;
```

```
const [result] = await connection.execute(insertQuery, [
```

```
  "varun rajput",
```

```
  "HR",
```

```
  30,
```

```
    });  
    console.log(`Inserted record with ID: ${result.insertId}`);  
    const selectQuery = `SELECT * FROM empdata`;   
    const [rows] = await connection.execute(selectQuery);  
    console.log("Employee Records:");  
    console.table(rows);  
    await connection.end();  
  } catch (error) {  
    console.error("Error connecting to the database:", error);  
  }  
}  
  
connectDatabase();
```

```
Connected to the database.  
Inserted record with ID: 0  
Employee Records:
```

| (index) | id | name | department | age |
|---------|------|----------------|------------|-----|
| 0 | null | 'varun rajput' | 'HR' | 30 |

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

application.

Index.js

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

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

```
const port = 8000;
```

```
app.listen(port, () => {
```

```
  console.log("Server is running");
```

```
});
```

Package.json

```
{
```

```
  "name": "question10",
```

```
  "version": "1.0.0",
```

```
  "description": "",
```

```
  "main": "index.js",
```

```
  "scripts": {
```

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

```
    "dev": "nodemon index.js",
```

```
"start": "echo \"Hello start.....\\",  
"user2": "echo \"Hello user2.....\\",  
"user3": "echo \"Hello user3.....\\",  
},  
"author": "",  
"license": "ISC",  
"dependencies": {  
  "express": "^4.19.2"  
}  
}
```

```
PS E:\assignnode\NodeJS Assignment\Question10> npm run start  
  
> question10@1.0.0 start  
> echo "Hello start....."  
  
"Hello start....."  
PS E:\assignnode\NodeJS Assignment\Question10> npm run user2  
  
> question10@1.0.0 user2  
> echo "Hello user2....."  
  
"Hello user2....."  
PS E:\assignnode\NodeJS Assignment\Question10> npm run user3  
  
> question10@1.0.0 user3  
> echo "Hello user3....."
```