1. Develop a web server with following functionalities:

- Serve static resources.

- Handle GET request.

- Handle POST request.

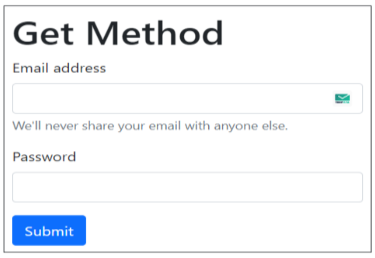
Server.js

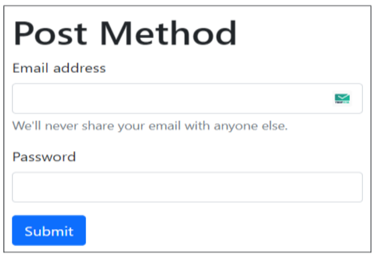
```
const http = require("http");
const url = require("url");
const fs = require("fs");
const static = require("node-static");

const fileServer = new static.Server("./public");
const server=http.createServer((req,res)=>{
    var url1=url.parse(req.url,true);

    if(url1.pathname=="/")
    {
        fs.readFile("./test.html","utf8",(err,data)=>{
          res.write(data)
          res.end()
        })

    }else if(url1.pathname=="/process" && req.method=="GET"){
        // console.log("hfghgf"+ req.url.query.email);
        res.write("Email : "+ url1.query.email+"\nPassword : " + url1.query.password);
        res.end();
    }else if(url1.pathname=="/process" && req.method=="POST"){
        var body='';
        req.on("data",chunk=>{
           body +=chunk;
        })
        req.on("end",()=>{
            res.write(body);

            res.end();

        });

    }else{

        res.end("Another Request is here");


    }

});


server.listen(3000, () => {

  console.log("server is running on port http://localhost:3000");

});
```

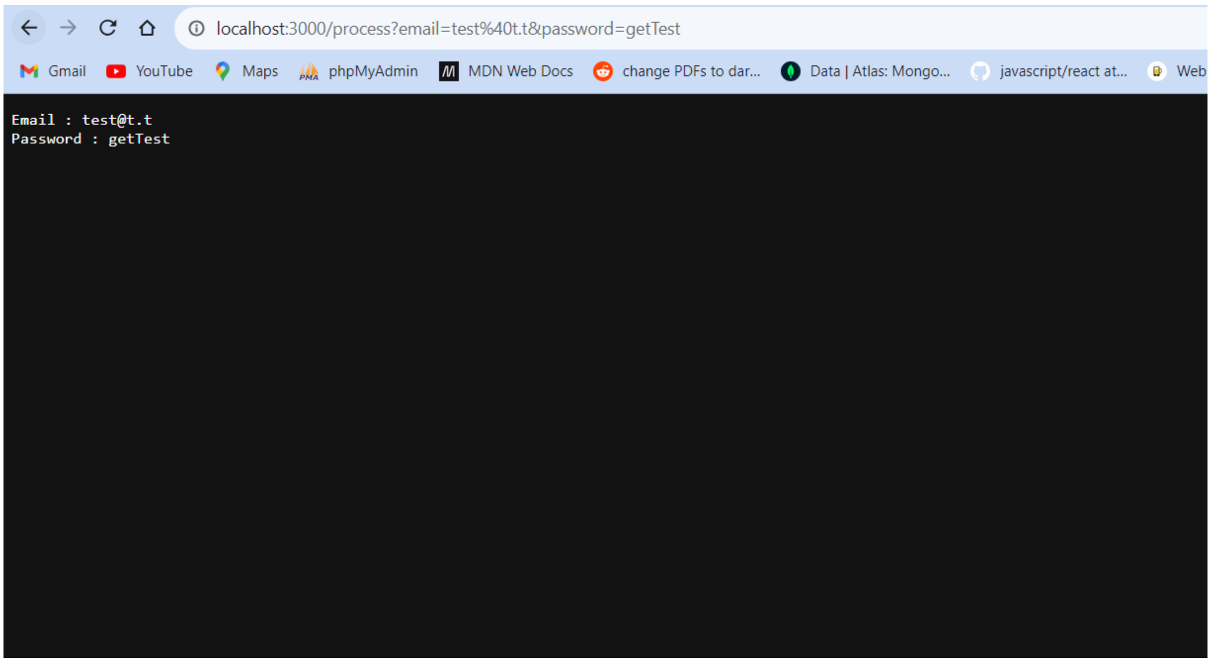
Index.html

```
<!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">


</head>

<body>


<div class="row">

    <div class="col">

    <form action="/process" method="GET" class="w-50 m-lg-5 border border-dark p-2" >

        <h1> Get Method</h1>
        <div class="mb-3">
          <label for="exampleInputEmail1" class="form-label">Email address</label>
          <input type="email" class="form-control" id="email" name="email" aria-
describedby="emailHelp">
          <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
        </div>
        <div class="mb-3">
          <label for="exampleInputPassword1" class="form-label">Password</label>
          <input type="password" class="form-control" id="password" name="password">
        </div>

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

    </div>
    <div class="col">
      <form action="/process" method="POST" class="w-50 m-lg-5 border border-dark p-2">
        <h1>Post Method</h1>
        <div class="mb-3">
          <label for="exampleInputEmail1" class="form-label">Email address</label>
          <input type="email" class="form-control" id="email" name="email" aria-
describedby="emailHelp">
          <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
        </div>
        <div class="mb-3">
          <label for="exampleInputPassword1" class="form-label">Password</label>
          <input type="password" class="form-control" id="password" name="password">
        </div>

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

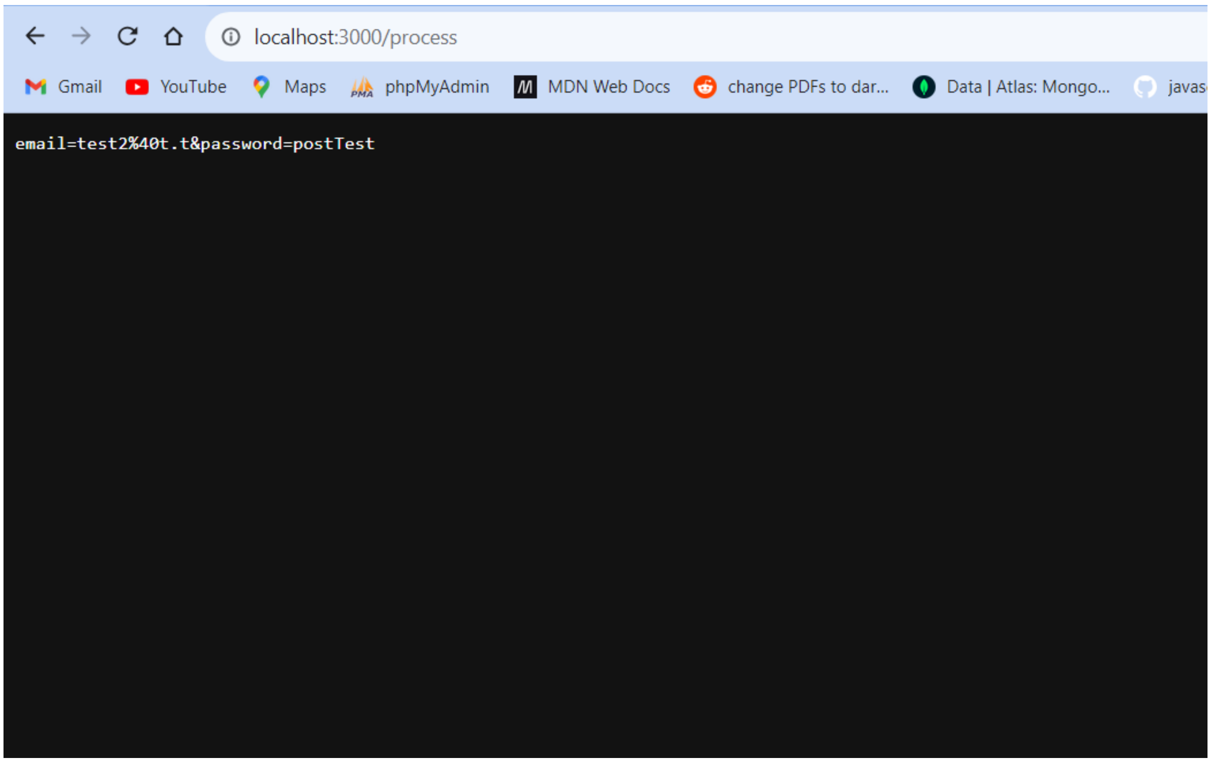
```
  </div>

  </div>

<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>


</body>

</html>
```







```
Email : test@t.t
Password : getTest
```



```
email=test2%40t.t&password=postTest
```

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

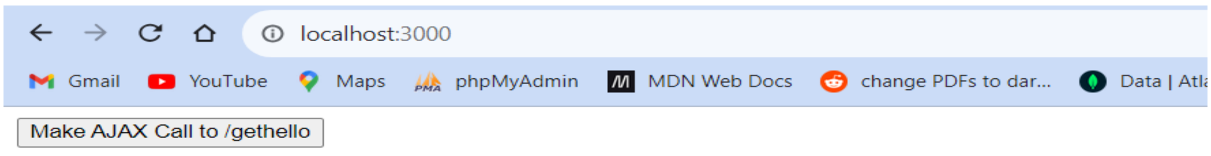
```
const express = require("express")
const app = express()
const fs = require("fs")

app.get("/",(req,res)=>{
   fs.readFile("./index.html","utf8",(err,data)=>{
      res.send(data)
   })
})

app.get("/gethello",(req,res)=>{
   fs.readFile("./index2.html","utf8",(err,data)=>{
      res.send(data)
   })
   // res.send(`<h1>hello nodeJs!!!</h1>`)
})

app.listen(3000)
```

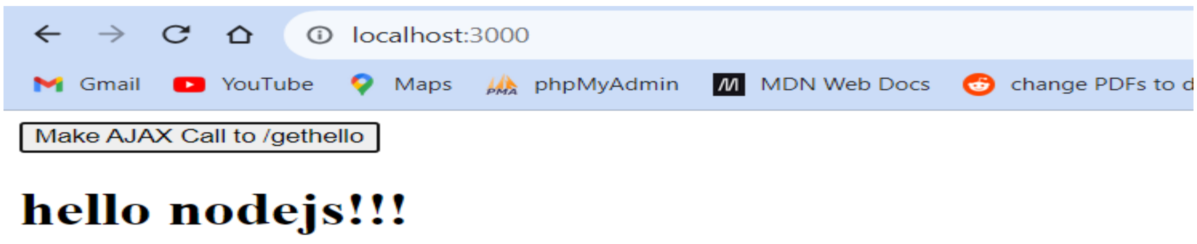
index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie=edge">
   <title>Document</title>
</head>
<body>
  <!-- <h1>hello nodejs!!!</h1> -->
  <button id="ajaxButton">Make AJAX Call to /gethello</button>
  <div id="ajaxResponse"></div>

</body>
<script>

document.getElementById("ajaxButton").addEventListener("click", () => {
    fetch("/gethello").then(response=>response.text()).then(data => {
       document.getElementById("ajaxResponse").innerHTML = data;
  }).catch(error => {
   console.error(error)
  });
});



</script>
</html>
```

Index2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie=edge">
   <title>Document</title>
</head>
<body>
  <h1>hello nodejs!!!</h1>

</body>
</html>
```





hello nodejs!!!

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

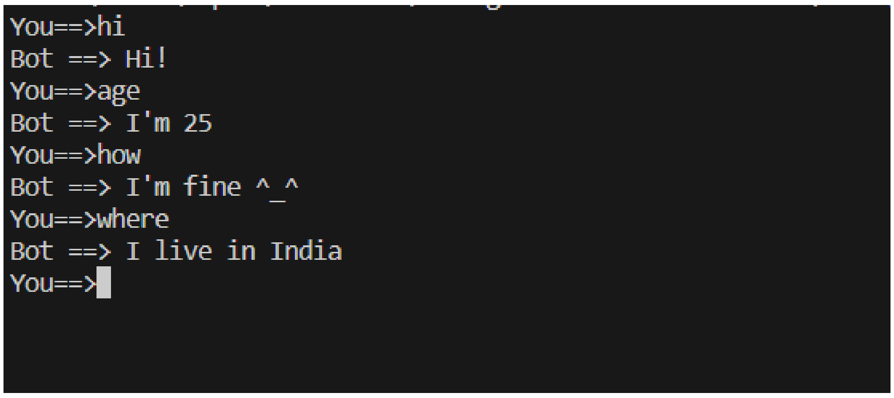
Server.js

```
var Chatbot = require('./q-3');

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);

});
```

Chatbot.js

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



```
You==>hi
Bot ==> Hi!
You==>age
Bot ==> I'm 25
You==>how
Bot ==> I'm fine ^_^
You==>where
Bot ==> I live in India
You==>
```

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

Server.js

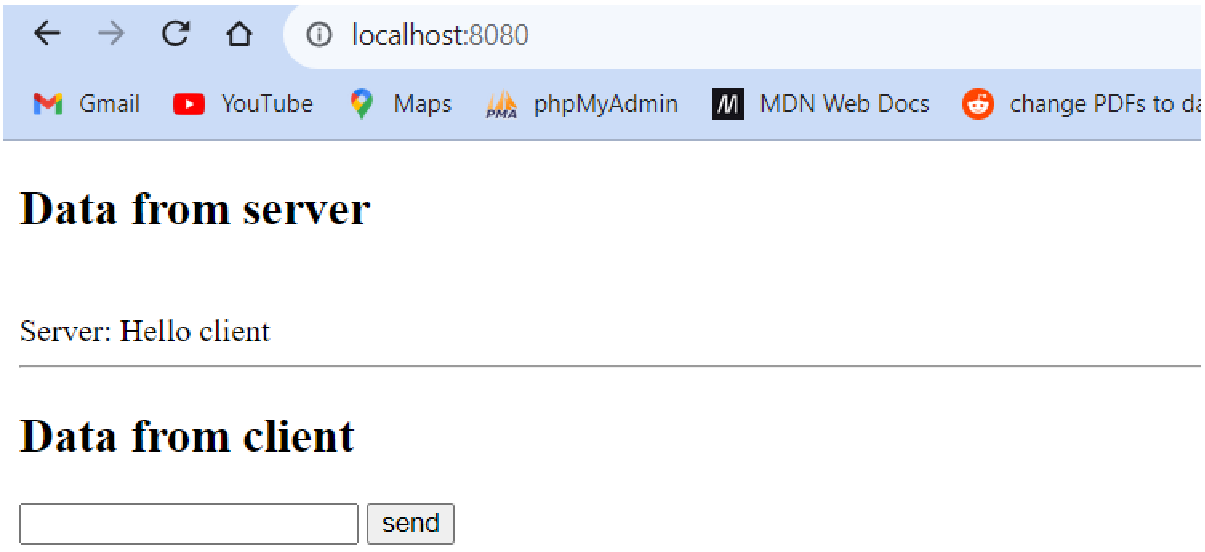
```
const WebSocket = require('ws')

var http = require('http');

var url = require('url');

var Chatbot = require('./q-3');


var st = require('node-static');


var fileServer = new st.Server('./public');


var httpserver = http.createServer(function(request, response)

{

        request.on('end', function () {

        var get = url.parse(request.url, true).query;

        fileServer.serve(request, response);

        }).resume();


}).listen(8080, function() {

  console.log((new Date()) +

   ' Server is listening on port 8080');

});
```

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

wss.on('connection', function(ws) {
  ws.send('Hello client')

  ws.on('message', message => {
    console.log(`Received message => ${message}`)
    // console.log(Chatbot.ChatbotReply(message))
    ws.send(Chatbot.ChatbotReply(message))
  })

})
```

Index.js

```
<!DOCTYPE html >
<html>
    <body>
<script language="javascript">

var ws = new WebSocket('ws://localhost:8080');

ws.addEventListener("message", function(e) {
  var msg = e.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="";
}
```

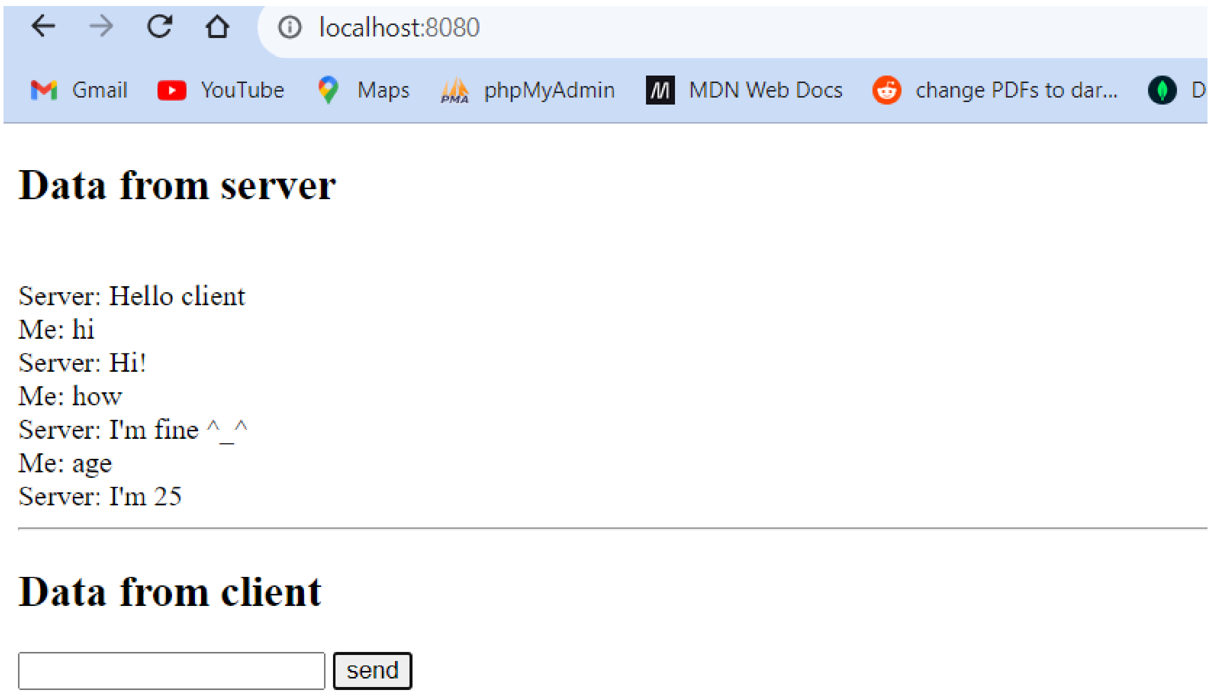
```
</script>
<h2>Data from server</h2>
    <div id="chatlog"></div>
<hr/>
<h2>Data from client</h2>
 <input type="text" id="message"  />
    <input type="button" id="b1" onclick="sendMessage()" value="send" />


  </body>
</html>
```



Data from server

Server: Hello client

Data from client



Data from server

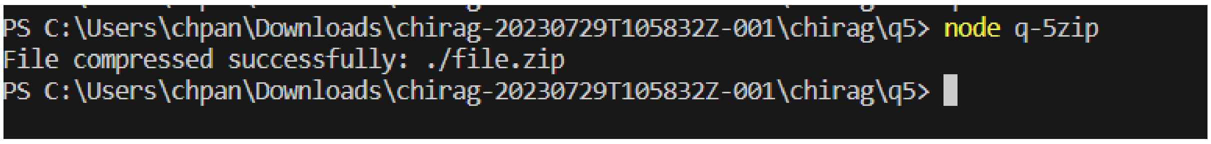
```
Server: Hello client
Me: hi
Server: Hi!
Me: how
Server: I'm fine ^_^
Me: age
Server: I'm 25
```

Data from client

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

Server.js

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

const zlib = require('zlib');


function compressFile(sourcePath, zipPath) {

 const readStream = fs.createReadStream(sourcePath);

 const writeStream = fs.createWriteStream(zipPath);


 readStream.pipe(zlib.createGzip()).pipe(writeStream);


 writeStream.on('finish', () => {

  console.log('File compressed successfully:', zipPath);

 });


 writeStream.on('error', (error) => {

  console.error('Error writing the ZIP file:', error);

 });
}
const sourceFile = './file.txt';
const zipFile = './file.zip';
compressFile(sourceFile, zipFile);
```

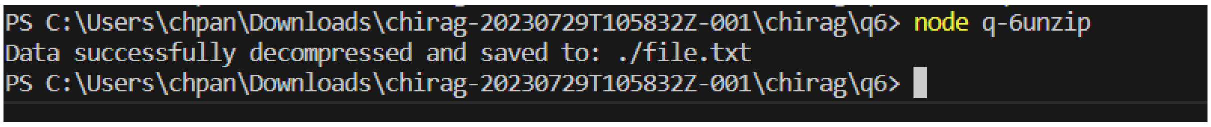


```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q5> node q-5zip
File compressed successfully: ./file.zip
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q5>
```

6. Write a program to extract a zip file.

Server.js

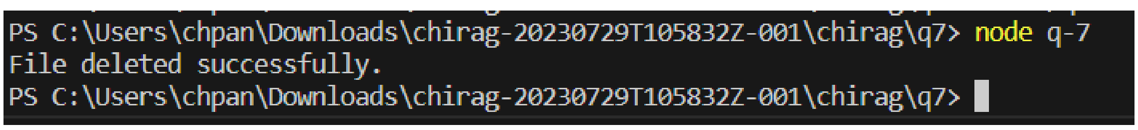
```
const fs = require('fs');
const zlib = require('zlib');

function decompressZlib(inputFilePath, outputFilePath) {
  const compressedData = fs.readFileSync(inputFilePath);
  zlib.unzip(compressedData, (error, decompressedData) => {
    if (error) {
      console.error('Error decompressing data:', error);
    } else {
      fs.writeFileSync(outputFilePath, decompressedData);
      console.log('Data successfully decompressed and saved to:', outputFilePath);
    }
  });
}

const compressedFilePath = './file.zip';
const decompressedFilePath = './file.txt';

decompressZlib(compressedFilePath, decompressedFilePath);
```



```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q6> node q-6unzip
Data successfully decompressed and saved to: ./file.txt
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q6>
```

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

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


function promisifiedUnlink(filePath) {
  return new Promise((resolve, reject) => {
    fs.unlink(filePath, (err) => {
      if (err) {
        reject(err);
      } else {
        resolve();
      }
    });
  });
}


async function deleteFile() {
    const filePath = './file.txt';


    try {


      await promisifiedUnlink(filePath);
      console.log('File deleted successfully.');
    } catch (err) {
      console.error('Error deleting file:', err);
    }
  }
  deleteFile();
```

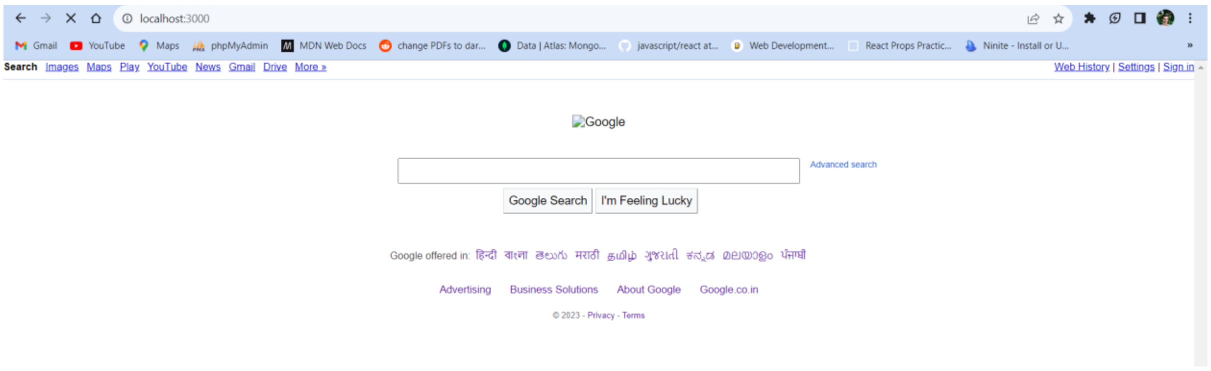


```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q7> node q-7
File deleted successfully.
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag\q7>
```

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

Server.js

```
const fetch = require('node-fetch');

const http = require('http');


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

  try {

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


    if (!response.ok) {

      throw new Error('Failed to fetch the Google homepage.');

    }


    const data = await response.text();

    console.log(data);

    res.write(data);

    res.end();

   } catch (error) {

    console.error('Error:', error.message);

   }



}).listen(3000,()=>{

  console.log(`server is listening on http://localhost:3000`);

});
```

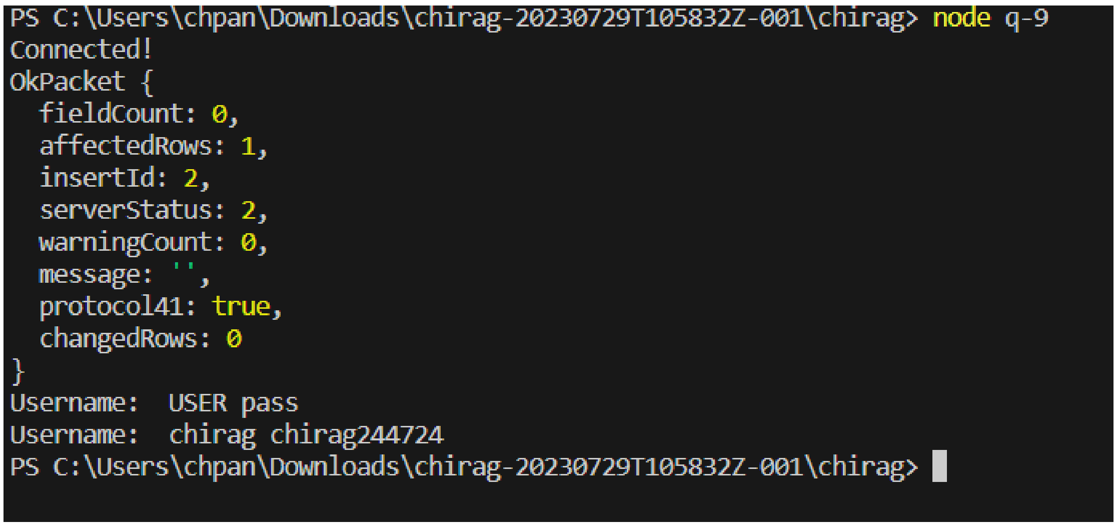

localhost:3000
Gmail YouTube Maps phpMyAdmin MDN Web Docs change PDFs to dar... Data | Atlas: Mongo... javascript/react at... Web Development... React Props Practic... Ninite - Install or U...
Search Images Maps Play YouTube News Gmail Drive More »
Web History | Settings | Sign in
Google
Advanced search
Google Search
I'm Feeling Lucky
Google offered in: हिन्दी বাংলা తెలుగు मराठी தமிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ
Advertising Business Solutions About Google Google.co.in
© 2023 - Privacy - Terms

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.

Server.js

```
const mysql = require('nodejs-mysql').default;


const config = {
    host: "localhost",
    user: "root",
    password: "root",
    database: "my"
}
const db = mysql.getInstance(config)


db.connect()
    .then(function(){
        console.log("Connected!");


        var sql = "INSERT INTO employee (username, password, firstname, lastname, email) VALUES
('chirag', 'chirag244724','fname1','lname1','a@b.com')";
        return db.exec(sql);


    }).then(function(res){
        console.log(res);
        return db.exec("SELECT * FROM employee");
```

```
  }).then(function(result){
    for (var i in result) {
      console.log('Username: ', result[i].username + " " +result[i].password);
    }
    process.exit(0);
  }).catch(function(err){
    console.log("Error");
    process.exit(0);
  })
```

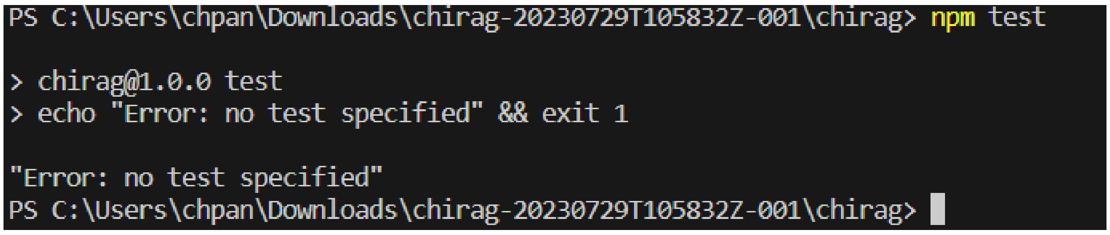


```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> node q-9
Connected!
OkPacket {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 2,
  serverStatus: 2,
  warningCount: 0,
  message: '',
  protocol41: true,
  changedRows: 0
}
Username:  USER pass
Username:  chirag chirag244724
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag>
```

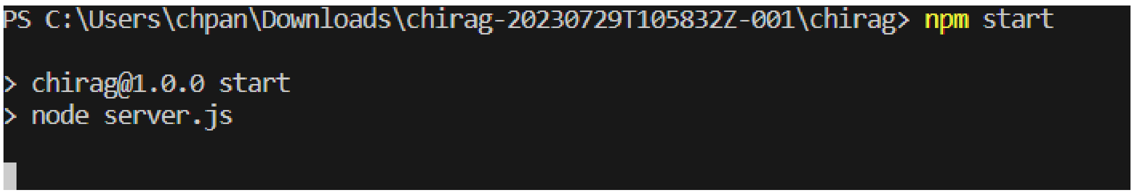
10. Set a server script, a test script and 3 user defined scripts in package.json file in your nodejs application.

Package.json

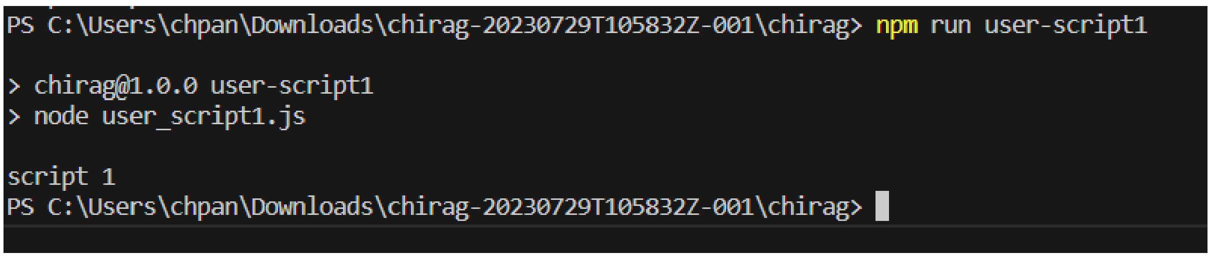
```
{
 "name": "chirag",
 "version": "1.0.0",
 "description": "",
 "main": "q-2.js",
 "scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "start": "node server.js",
  "user-script1": "node user_script1.js",
  "user-script2": "node user_script2.js",
   "user-script3": "node user_script3.js"
 },
 "keywords": [],
 "author": "",
 "license": "ISC",
 "dependencies": {
  "express": "^4.18.2",
  "node-fetch": "^2.6.0",
  "node-static": "^0.7.11",
  "nodejs-mysql": "^0.1.3",
  "request-promise": "^4.2.6",
  "ws": "^8.13.0"
 }
}
```



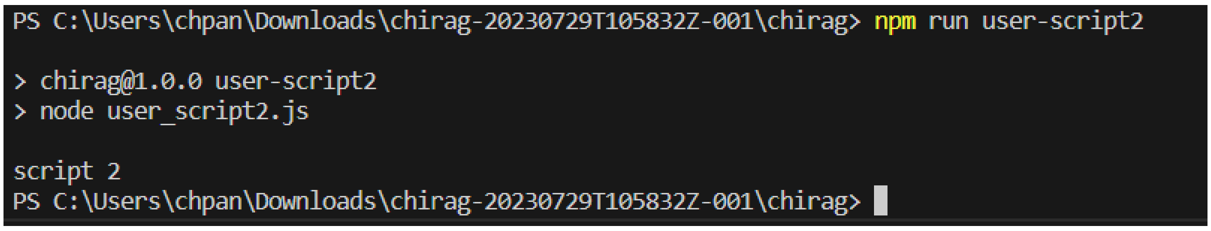
```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> npm test

> chirag@1.0.0 test
> echo "Error: no test specified" && exit 1

"Error: no test specified"
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag>
```



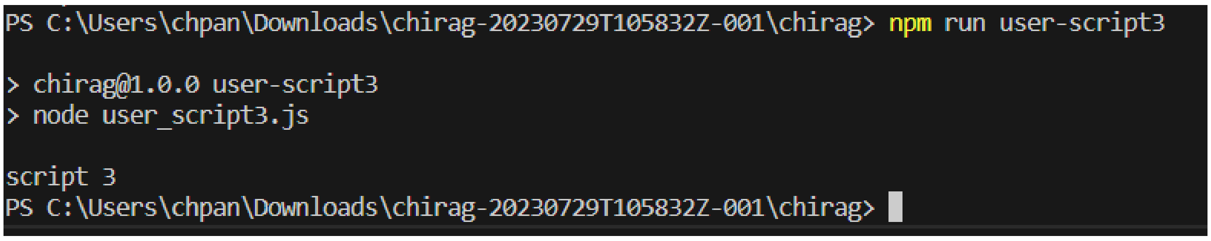
```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> npm start

> chirag@1.0.0 start
> node server.js
```



```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> npm run user-script1

> chirag@1.0.0 user-script1
> node user_script1.js

script 1
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag>
```



```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> npm run user-script2

> chirag@1.0.0 user-script2
> node user_script2.js

script 2
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag>
```

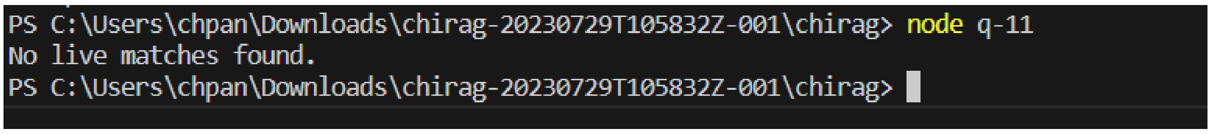


```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> npm run user-script3

> chirag@1.0.0 user-script3
> node user_script3.js

script 3
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag>
```

11. Develop an application to show live cricket score.

Server.js

```
const request = require('request-promise');


async function getLiveCricketScores() {
 try {
  const apiKey = '13cf787f-72cd-41ca-9e2f-3d711cb26c6e'; // Replace this with your actual API key
  const apiUrl = `https://cricapi.com/api/matches?apikey=${apiKey}`;


  const response = await request(apiUrl, { json: true });


  if (response.error) {
   throw new Error(response.error);
  }


  const matches = response.matches;
  if (!matches || matches.length === 0) {
   console.log('No live matches found.');
   return;
  }


  console.log('Live Cricket Scores:');
  console.log('---------------------');
  matches.forEach((match) => {
   const { team1, team2, score } = match;
   console.log(`${team1} vs ${team2}: ${score}`);
  });
  } catch (error) {
    console.error('Error:', error.message);
  }
}

getLiveCricketScores();
```



```
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> node q-11
No live matches found.
PS C:\Users\chpan\Downloads\chirag-20230729T105832Z-001\chirag> 
```