SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN (AUTONOMOUS)

VISHNUPUR, BHIMAVARAM - 534 202

**(Permanently Affiliated to JNTUK, Kakinada,
Accredited by NBA & NAAC with A+ Grade)**

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

LABORATORY RECORD

VISHNU
UNIVERSAL LEARNING

NAME :

REGD. NO :

CLASS :

LABORATORY :

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN
(AUTONOMOUS)
VISHNUPUR, BHIMAVARAM - 534 202
(Permanently Affiliated to JNTUK, Kakinada,
Accredited by NBA & NAAC with A+ Grade)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

Certificate

Certified that this is a bonafide record of practical work done by Ms.____ Regd.No.___________________ of II / II B.Tech in the "*Web Application Development Laboratory*" during the year 2023-2024

Date **Staff Member In-Charge** **Head of the Department**

Submitted for the Practical Examination held on: ________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(AUTONOMOUS)

VISHNUPUR, BHIMAVARAM - 534 202

INDEX

S. NO.	**Date**	**Experiment**	**Page**	**Marks**
		Total Avg. Marks		

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(AUTONOMOUS)

VISHNUPUR, BHIMAVARAM - 534 202

INDEX

S. NO.	**Date**	**Experiment**	**Page**	**Marks**
		Total Avg. Marks		

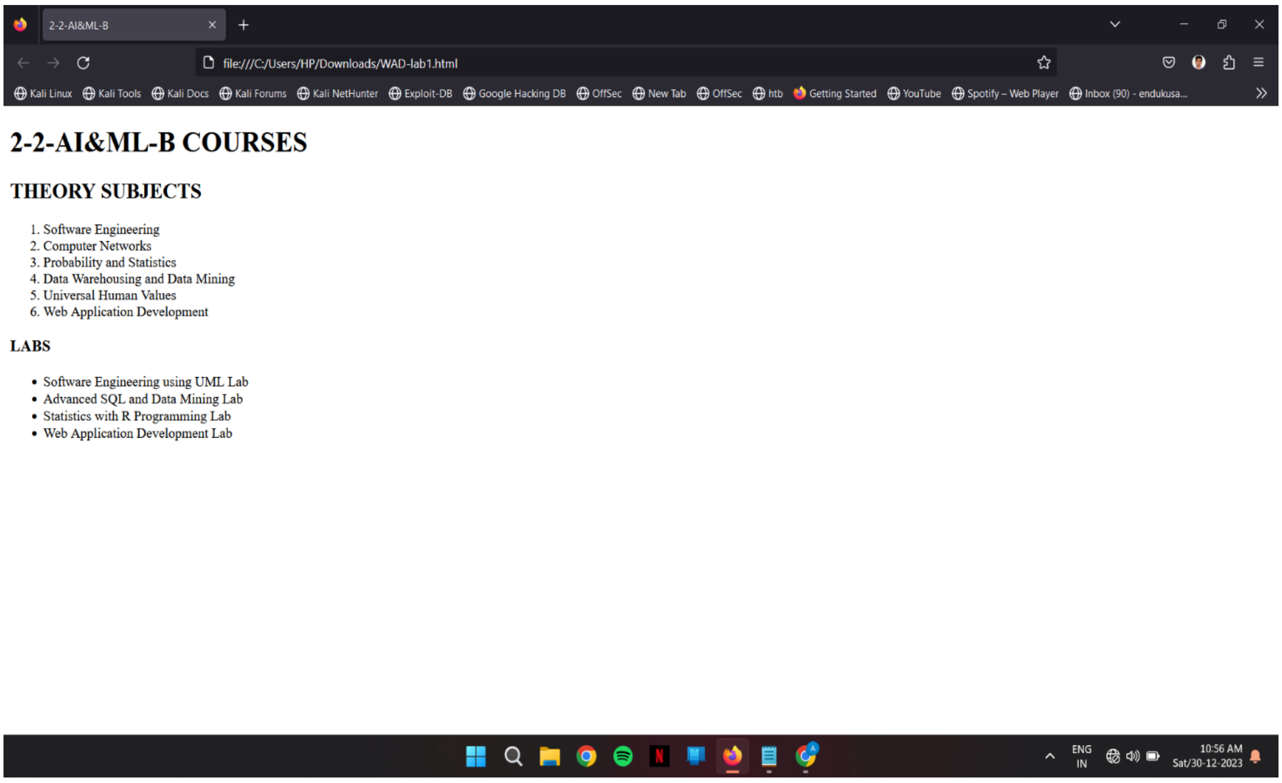
EXPERIMENT – 1:

Develop a static web page using HTML Tags , List Tags , Image Tags.

Code:

```
<!DOCTYPE html>

<html>

<head>

<title> 2-2-AI&ML-B</title>

</head>

<body>

<h1>2-2-AI&ML-B COURSES</hi>

<h2>THEORY SUBJECTS</h2>

<ol>

        <li>Software Engineering</li>

        <li>Computer Networks</li>

        <li>Probability and Statistics</li>

        <li>Data Warehousing and Data Mining</li>

        <li>Universal Human Values</li>

        <li>Web Application Development</li>

</ol>

<h3>LABS</h3>

<ul>

        <li>Software Engineering using UML Lab</li>

        <li>Advanced SQL and Data Mining Lab</li>

        <li>Statistics with R Programming Lab</li>

        <li>Web Application Development Lab</li>

</ul>

</body>

</html>
```

OUTPUT :



2-2-AI&ML-B

file:///C:/Users/HP/Downloads/WAD-lab1.html

2-2-AI&ML-B COURSES

THEORY SUBJECTS

1. Software Engineering
2. Computer Networks
3. Probability and Statistics
4. Data Warehousing and Data Mining
5. Universal Human Values
6. Web Application Development

LABS

- Software Engineering using UML Lab
- Advanced SQL and Data Mining Lab
- Statistics with R Programming Lab
- Web Application Development Lab

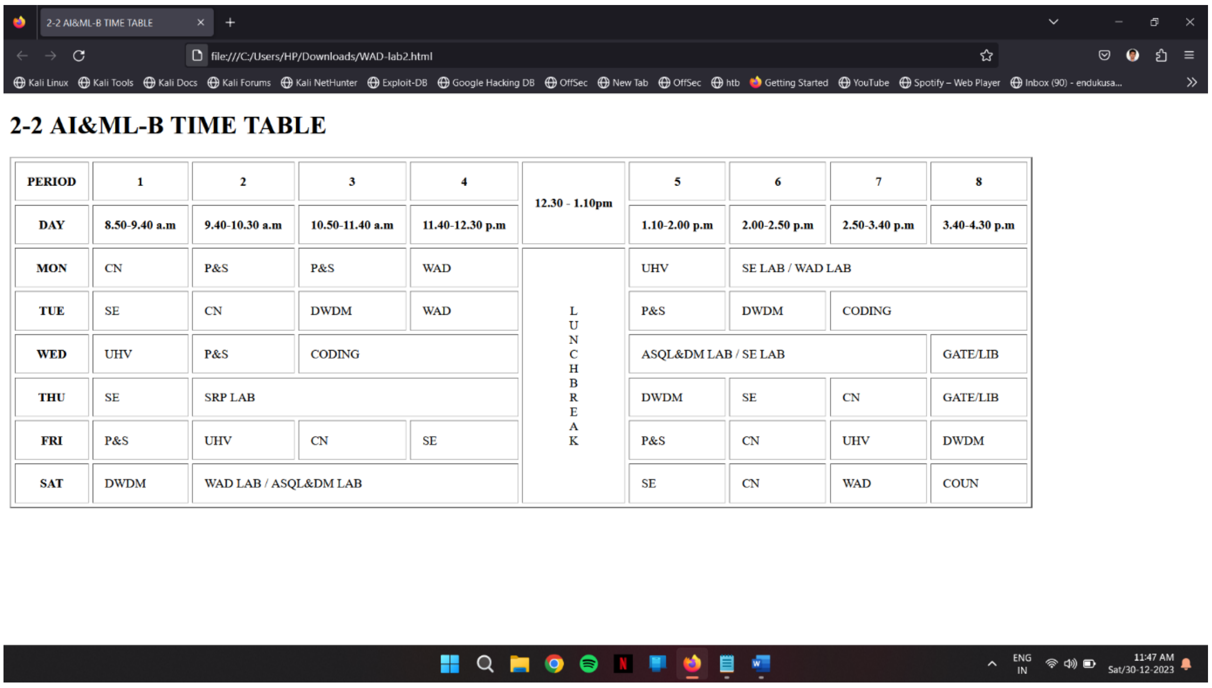
EXPERIMENT – 2:

Demonstrate table tag to create different orientation of table in static web page.

CODE:

```
<!DOCTYPE html>

<html>

<head>

<title> 2-2 AI&ML-B TIME TABLE</title>

</head>

<body>

<h1>2-2 AI&ML-B TIME TABLE </h1>

<table border="2" cellspacing=”5” cellpadding=”15”>

<tr>

        <th>PERIOD</th>

        <th>1</th>

        <th>2</th>

        <th>3</th>

        <th>4</th>

        <th rowspan="2"> 12.30 - 1.10pm</th>

        <th>5</th>

        <th>6</th>

        <th>7</th>

        <th>8</th>

</tr>

<tr>

        <th>DAY</th>

        <th>8.50-9.40 a.m</th>

        <th>9.40-10.30 a.m</th>

        <th>10.50-11.40 a.m</th>
        <th>11.40-12.30 p.m</th>

        <th>1.10-2.00 p.m</th>

        <th>2.00-2.50 p.m</th>

        <th>2.50-3.40 p.m</th>

        <th>3.40-4.30 p.m</th>

</tr>

<tr>

        <th>MON</th>

        <td>   CN</td>

        <td >   P&S</td>

        <td> P&S</td>

        <td>   WAD</td>

        <td
rowspan="6"><center>L<br>U<br>N<br>C<br>H<br>B<br>R<br>E<br>A<br>K<br></center> </td>

        <td>   UHV</td>

        <td colspan="3">   SE LAB / WAD LAB</td>

</tr>

<tr>

        <th>TUE</th>

        <td>SE</td>

        <td>CN</td>

        <td>DWDM</td>

        <td>WAD</td>

        <td>P&S</td>

        <td>DWDM</td>

        <td colspan="2">CODING</td>

</tr>

<tr>

        <th>WED</th>

        <td>UHV</td>

        <td>P&S</td>
        <td colspan="2">CODING</td>
        <td colspan="3">ASQL&DM LAB / SE LAB</td>
        <td>GATE/LIB</td>
</tr>
<tr>
        <th>THU</th>
        <td>SE</td>
        <td colspan="3">SRP LAB</td>
        <td>DWDM</td>
        <td>SE</td>
        <td>CN</td>
        <td>GATE/LIB</td></tr>
<tr><th>FRI</th>
        <td>P&S</td>
        <td>UHV</td>
        <td>CN</td>
        <td>SE</td>
        <td>P&S</td>
        <td>CN</td>
        <td>UHV</td>
        <td>DWDM</td></tr>
<tr><th>SAT</th>
        <td>DWDM</td>
        <td colspan="3">WAD LAB / ASQL&DM LAB</td>
        <td>SE</td>
        <td>CN</td>
        <td>WAD</td>
        <td>COUN</td>
</tr>
</table>
</body></html>
```

OUTPUT :



2-2 AI&ML-B TIME TABLE

PERIOD	1	2	3	4	12.30 - 1.10pm	5	6	7	8
DAY	8.50-9.40 a.m	9.40-10.30 a.m	10.50-11.40 a.m	11.40-12.30 p.m		1.10-2.00 p.m	2.00-2.50 p.m	2.50-3.40 p.m	3.40-4.30 p.m
MON	CN	P&S	P&S	WAD	LUNCH BREAK	UHV	SE LAB / WAD LAB		
TUE	SE	CN	DWDM	WAD		P&S	DWDM	CODING	
WED	UHV	P&S	CODING			ASQL&DM LAB / SE LAB			GATE/LIB
THU	SE	SRP LAB				DWDM	SE	CN	GATE/LIB
FRI	P&S	UHV	CN	SE		P&S	CN	UHV	DWDM
SAT	DWDM	WAD LAB / ASQL&DM LAB				SE	CN	WAD	COUN

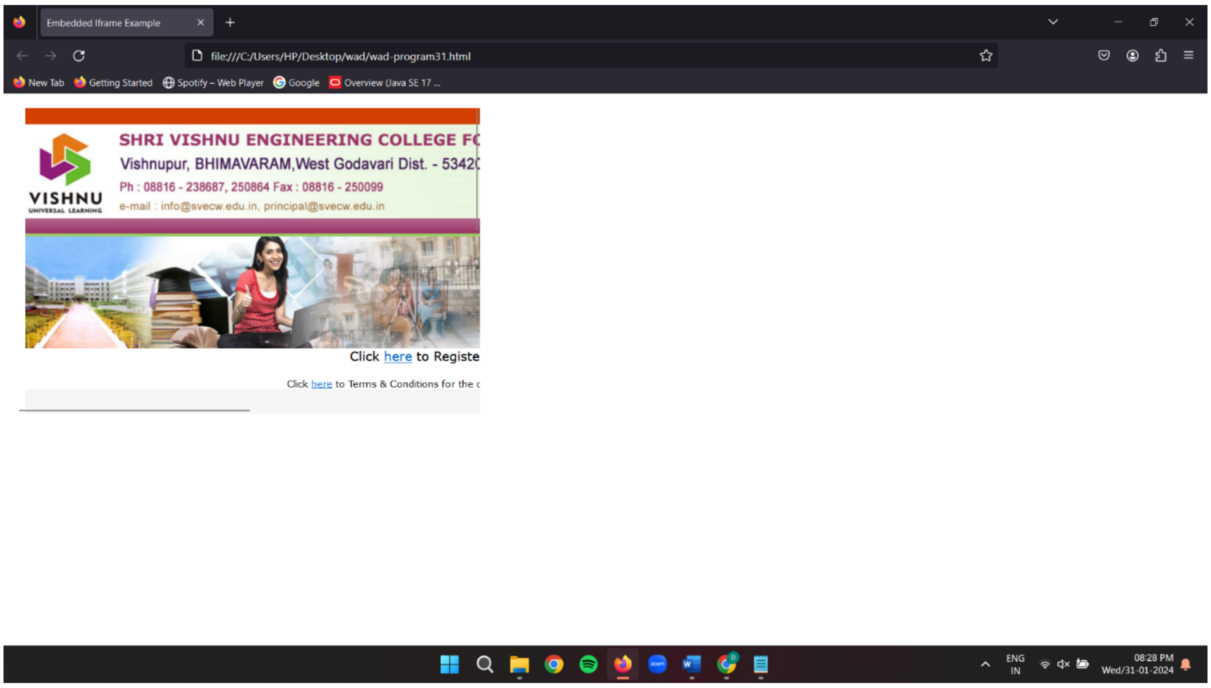
EXPERIMENT - 3

Develop static web page having different partitions using iframes.

Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

   <meta charset="UTF-8">

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

   <title>Embedded Iframe Example</title>

</head>

<body>


<iframe src="https://svecw.ac.in/Default.aspx?ReturnUrl=%2f" width="600" height="400"
frameborder="0" allowfullscreen></iframe>


</body>

</html>
```

OUTPUT :



SHRI VISHNU ENGINEERING COLLEGE F

Vishnupur, BHIMAVARAM,West Godavari Dist. - 5342

Ph : 08816 - 238687, 250864 Fax : 08816 - 250099

e-mail : info@svecw.edu.in, principal@svecw.edu.in

Click here to Registe

Click here to Terms & Conditions for the

EXPERIMENT – 4

Develop a web page to demonstrate CSS properties.

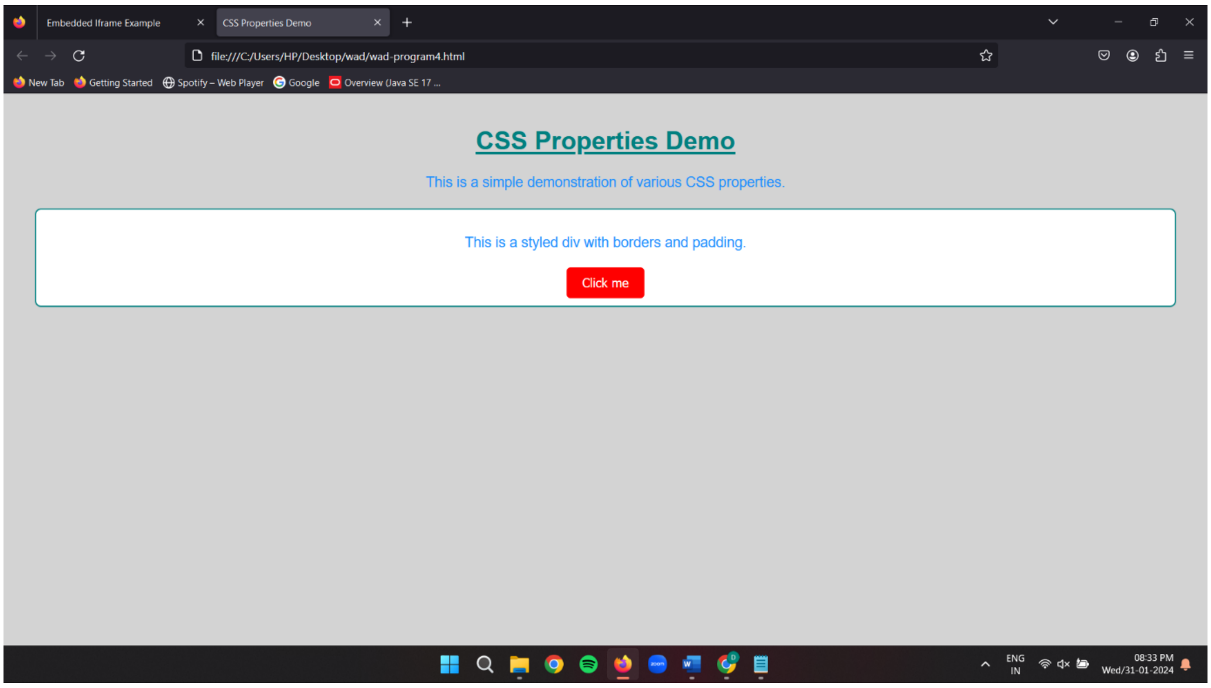
Code :

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>CSS Properties Demo</title>
 <style>
  /* Styling the body */
  body {
   font-family: Arial, sans-serif;
   background-color: lightgrey;
   color: #333;
   margin: 0;
   padding: 20px;
   text-align: center;
  }
  /* Styling a heading */
  h1 {
   color: #008080;
   text-decoration: underline;
  }
  /* Styling a paragraph */
  p {
   font-size: 18px;
   line-height: 1.5;
```

```
            color:dodgerblue;
    }

    /* Styling a div */
    .box {
      border: 2px solid #008080;
      padding: 10px;
      margin: 20px;
      background-color: #fff;
      border-radius: 8px;
    }
    /* Styling a button */
    button {
      background-color: red;
      color: white;
      padding: 10px 20px;
      font-size: 16px;
      border: none;
      border-radius: 5px;
      cursor: pointer;
    }
  </style>
</head>
<body>
  <h1>CSS Properties Demo</h1>
  <p>This is a simple demonstration of various CSS properties.</p>
  <div class="box">
    <p>This is a styled div with borders and padding.</p>
    <button>Click me</button>
  </div>
</body>
```

```
</html>
```

OUTPUT :



CSS Properties Demo

This is a simple demonstration of various CSS properties.

This is a styled div with borders and padding.

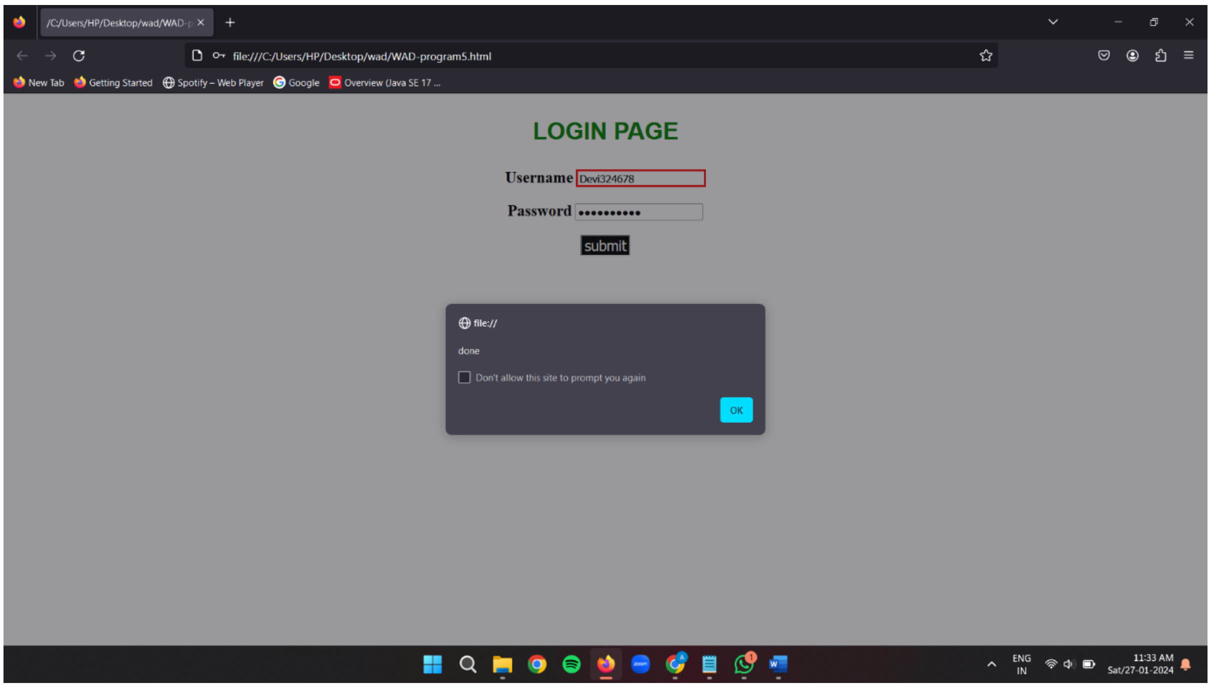
EXPERIMENT -5

Design a dynamic web page with validation of various form elements using JavaScript regular expression.

Code :

```
<!DOCTYPE html>

<html>

<head>

        <style type="text/css">

                .form-container{

                        text-align: center;

                }

                p{

                        font-family: sans-serif;

                        font-weight: bolder;

                        font-size: 30px;

                        color: green;

                }

        </style>

<script  type="text/javascript">


function validateusername()

{

let  regEx1=/^[A-Za-z0-9]{10}$/;

let x=document.getElementById("username").value;

let x1=document.getElementById("username");

if(regEx1.test(x))

{

window.alert("done");

return true;
}
else {
x1.style.border="solid red 3px";
window.alert("Enter valid username");
return false;
}
}
</script>
</head>
<body>
        <div class="form-container">
                <p>LOGIN PAGE</p>
<form action="#" >
        <label for="username" style="font-weight: bold;font-size: 20px;">Username</label>
<input type="text" id="username" placeholder =" Username"pattern="[A-Za-z0-9]{10}" title="10
characters along with numbers" required/>
<br><br>
<label for="p1" style="font-weight: bold;font-size: 20px;">Password</label>
<input type="password" id="p1" placeholder="password" />
<br><br>
<input type="button" style="font-size: 18px;color: white;background-color: black; border:"
value="submit" onclick="return validateusername()" />
</form>
</div>
</body>
```

OUTPUT :


LOGIN PAGE

Username Devi324678

Password ••••••••••

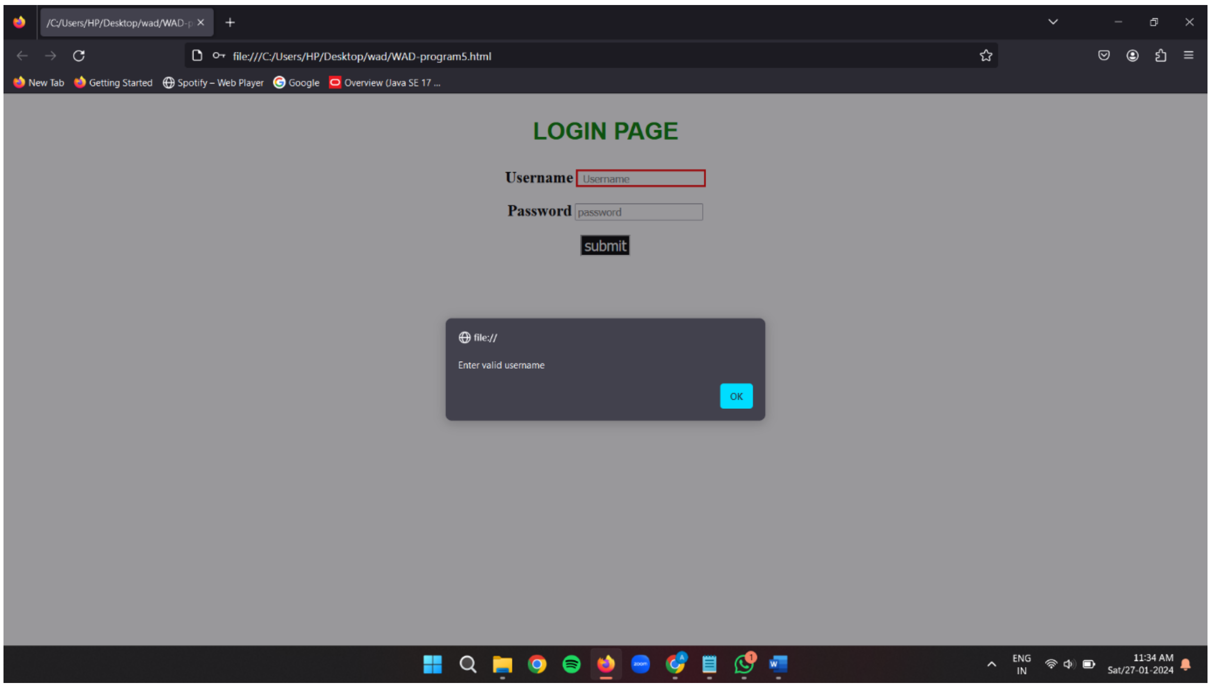
submit

file://

done

Don't allow this site to prompt you again

OK


LOGIN PAGE

Username Username

Password password

submit

file://

Enter valid username

OK

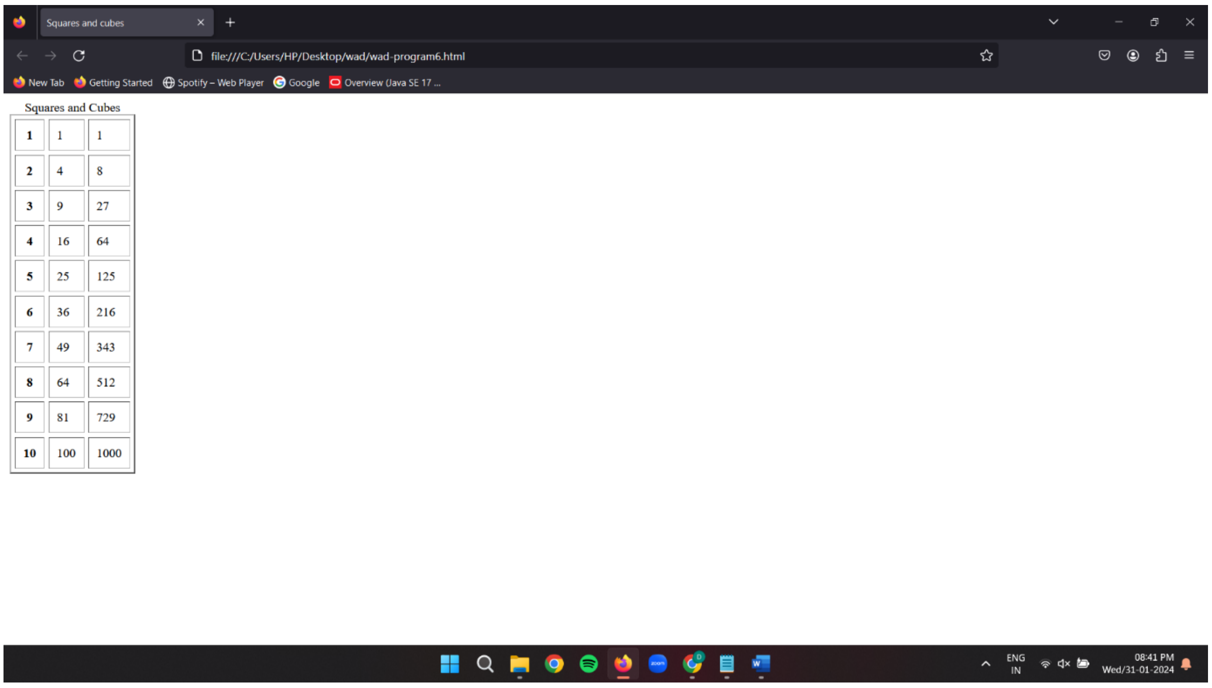
EXPERIMENT -6

Write a JavaScript that calculates the squares and cubes of the numbers from 0 to 10 and outputs HTML text that displays the resulting values in an HTML table format.

Code :

```
<!DOCTYPE html>
<html>
<head>
<title> Squares and cubes </title>
<script type="text/javascript">
function squarescubes(){
   let h;
   document.write("<table border=2 cellpadding=10 cellspacing=5><caption>Squares and
Cubes</caption>");

   for(h = 1; h <= 10; h++) {
      document.write("<tr><th>" + h + "</th>");
      document.write("<td>" + h * h + "</td><td>" + h * h * h + "</td></tr>");
   }

   document.write("</table>");
}
</script>
</head>
<body onload="squarescubes()">

<h1>Squares and Cubes from Number 1 to 10</h1>

</body>
</html>
```

OUTPUT :



Squares and Cubes

1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

EXPERIMENT – 7:

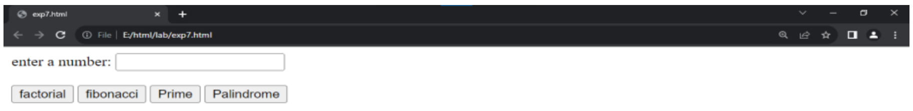
Design an HTML having a text box and four buttons for Factorial, Fibonacci, Prime, and Palindrome. When a button is pressed an appropriate javascript function should be called to display 1. Factorial of that number 2. Fibonacci series up to that number 3. Prime numbers up to that number Is it palindrome or not.

Code :

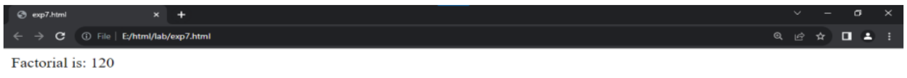
```
<!-- fibonacci and factorial using javascript program -->
<!DOCTYPE html>
<html>
  <head>
    <script type = "text/javascript">
      function fact()
      {
        var y = document.getElementById("no").value;
        var h,x = 1;
        for(h=1;h<=y;h++)
        x = x*h;
      document.getElementById("l1").innerHTML = x;
      }
      function fibonacci()
      {
        let y = document.getElementById("no").value
        let a=0,b=1,c,k=2;
        let fmsg = "Fibonacci series is : "+a+""+b;
        while(k<y)
        {
          k++;
          c = a+b;
          a = b;
          b = c;
```

```
                fmsg = fmsg+" "+c;
            }
            document.getElementById("l1").innerHTML=" "+fmsg;
        }
    </script>
  </head>
  <body>
      <h1>Fibonacci series</h1>
      <label style="color:rgb(255, 0, 217);" id="l1"></label><br>
      <label>Enter number and press button to get the output</label>
      <input type="number" id="no" name="no" placeholder="Enter a number">
      <br>
      <input type="button" onclick="fact()" value="FACTORIAL">&nbsp;&nbsp;
      <input type="button" onclick="fibonacci()" value="FIBONACCI">
  </body>
</html>
```

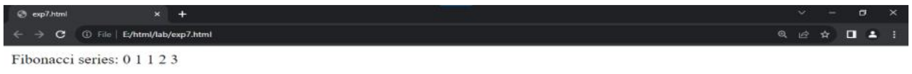
OUTPUT :



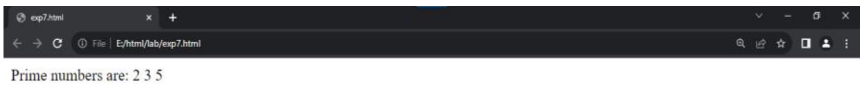
```
enter a number: [            ]
[factorial] [fibonacci] [Prime] [Palindrome]
```



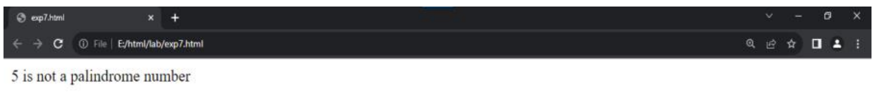
```
Factorial is: 120
```



```
Fibonacci series: 0 1 1 2 3
```



```
Prime numbers are: 2 3 5
```



```
5 is not a palindrome number
```

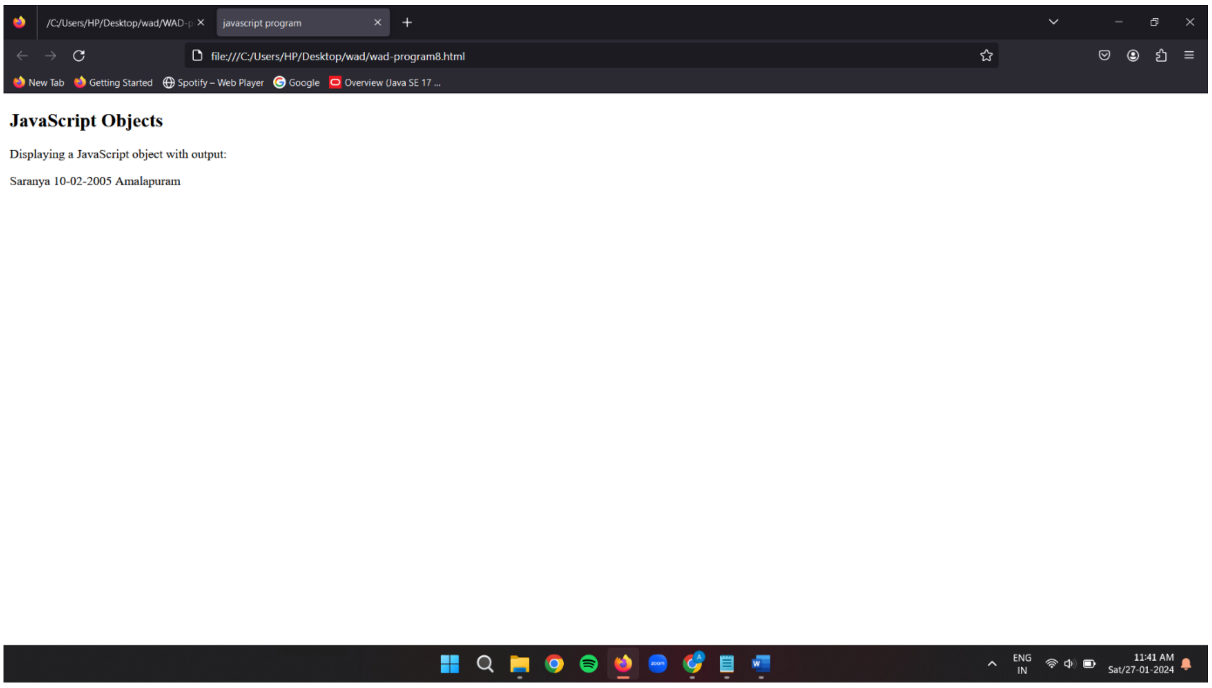
EXPERIMENT – 8:

Write a Java script code to demonstrate the objects.

Code :

```
<!DOCTYPE html>
<html>
<head>
<title> javascript program </title>
</head>
<body>

<h2>JavaScript Objects</h2>
<p>Displaying a JavaScript object with output:</p>

<p id="demo"></p>

<script>
const person = {
  name: "Saranya",
  DOB : "10-02-2005",
  city: "Amalapuram",

};

document.getElementById("demo").innerHTML = person.name+" "+person.DOB+" " + person.city;
</script>

</body>
</html>
```

OUTPUT :



```
JavaScript Objects

Displaying a JavaScript object with output:

Saranya 10-02-2005 Amalapuram
```

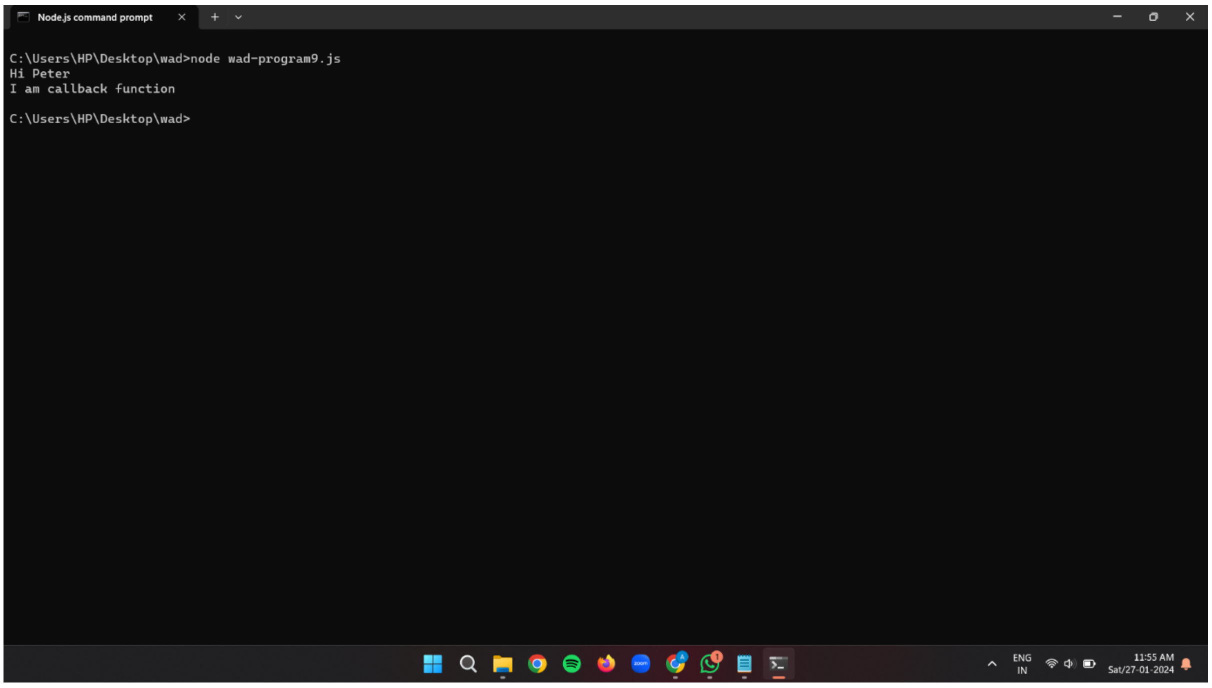
EXPERIMENT – 9 :

Write a java script code to demonstrate the callback function.

Code :

```
function greet(name, callback) {
   console.log('Hi' + ' ' + name);
   callback();
}

// callback function
function callMe() {
   console.log('I am callback function');
}

// passing function as an argument
greet('Peter', callMe);
```

OUTPUT :

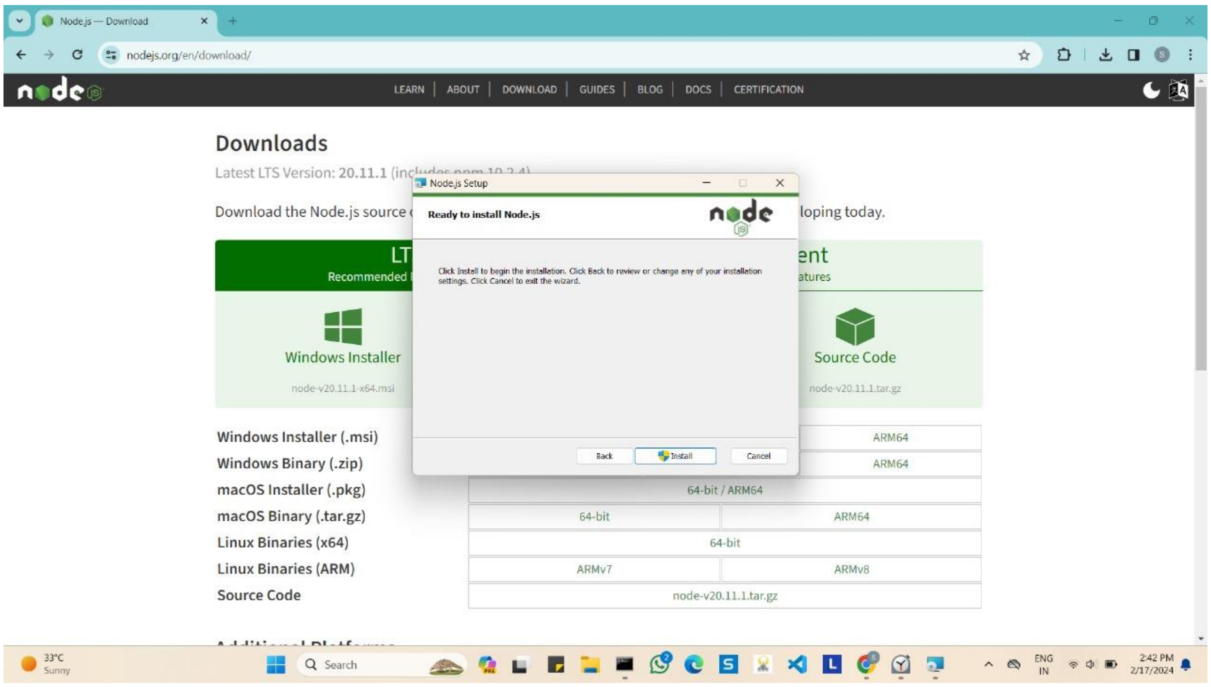


```
C:\Users\HP\Desktop\wad>node wad-program9.js
Hi Peter
I am callback function

C:\Users\HP\Desktop\wad>
```

EXPERIMENT-10:

Demonstrate the installation of NODE.JS .

OUTPUT :


Node.js — Download
nodejs.org/en/download/
LEARN | ABOUT | DOWNLOAD | GUIDES | BLOG | DOCS | CERTIFICATION
Back
Install
Cancel

Downloads

Latest LTS Version: 20.11.1

Download the Node.js source … loping today.

Node.js Setup

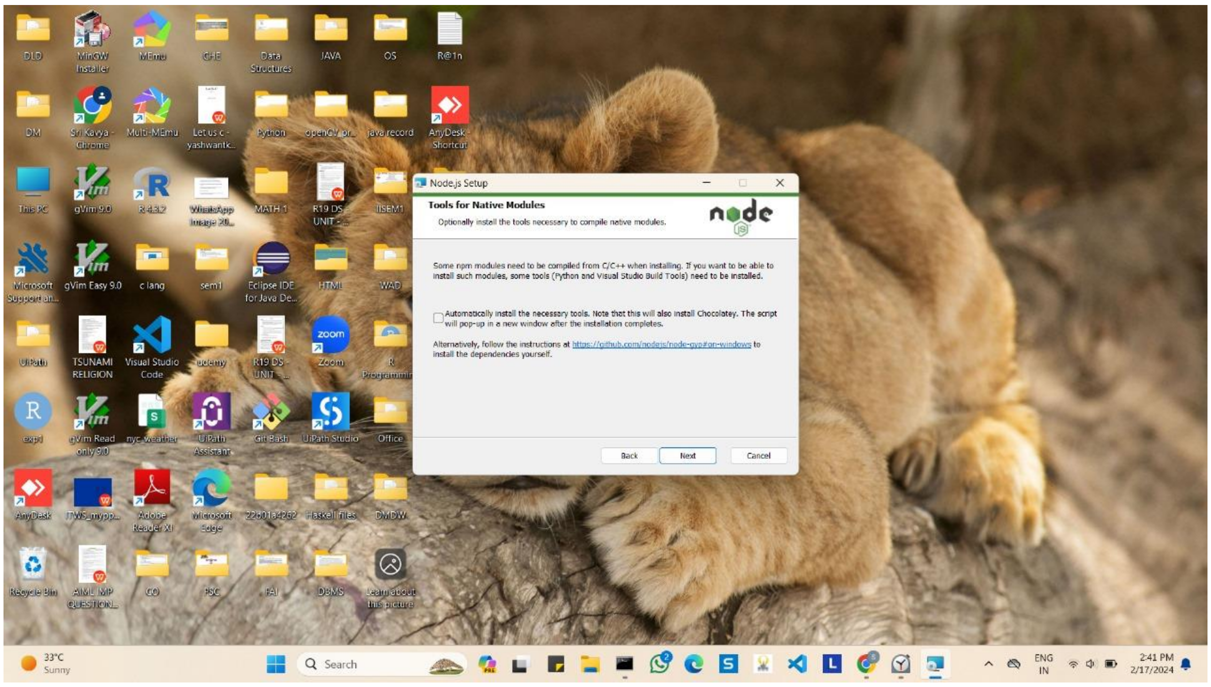
Ready to install Node.js

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Windows Installer
node-v20.11.1-x64.msi

Source Code
node-v20.11.1.tar.gz

Windows Installer (.msi)		ARM64
Windows Binary (.zip)		ARM64
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v20.11.1.tar.gz	



Node.js Setup

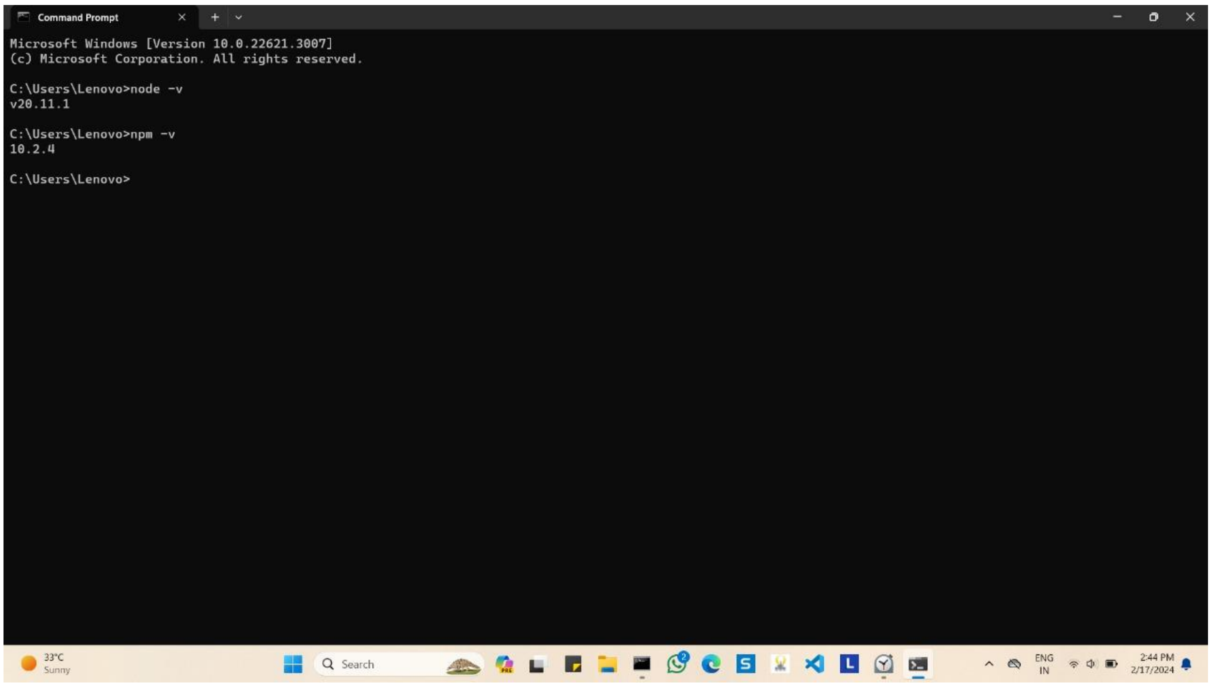
Tools for Native Modules

Optionally install the tools necessary to compile native modules.

Some npm modules need to be compiled from C/C++ when installing. If you want to be able to install such modules, some tools (Python and Visual Studio Build Tools) need to be installed.

☐ Automatically install the necessary tools. Note that this will also install Chocolatey. The script will pop-up in a new window after the installation completes.

Alternatively, follow the instructions at https://github.com/nodejs/node-gyp#on-windows to install the dependencies yourself.



```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>node -v
v20.11.1

C:\Users\Lenovo>npm -v
10.2.4

C:\Users\Lenovo>
```

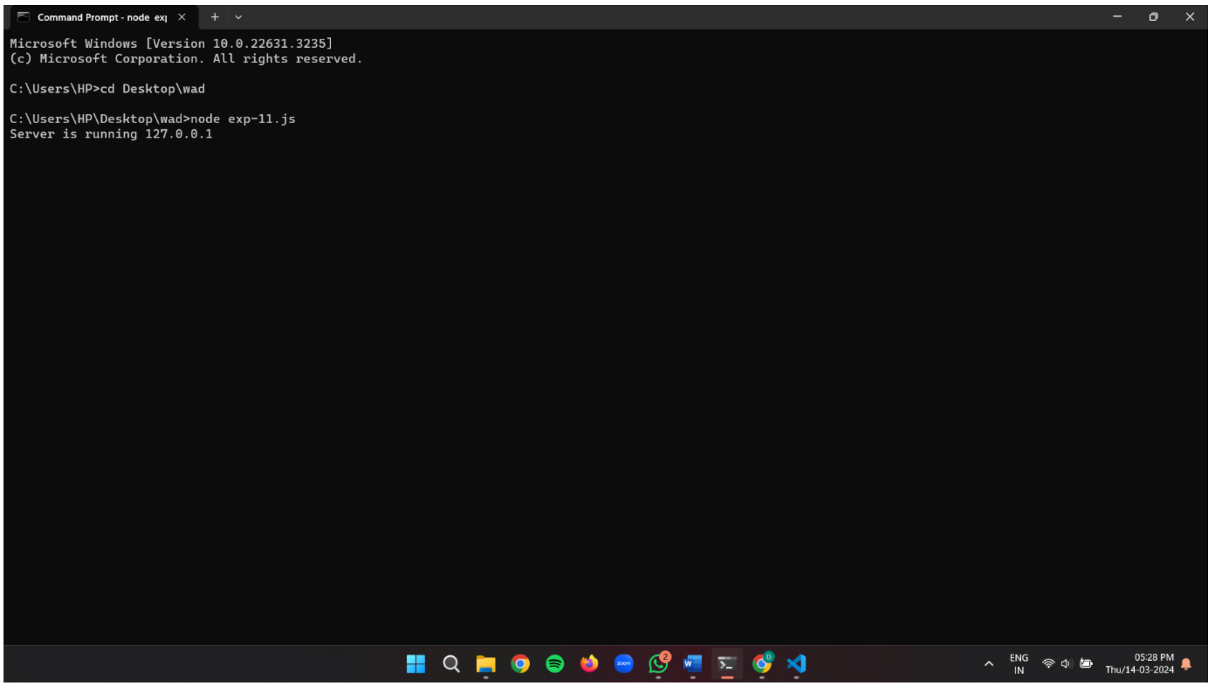
EXPERIMENT – 11:

Demonstrate the process of importing NPM Modules, Core Modules.

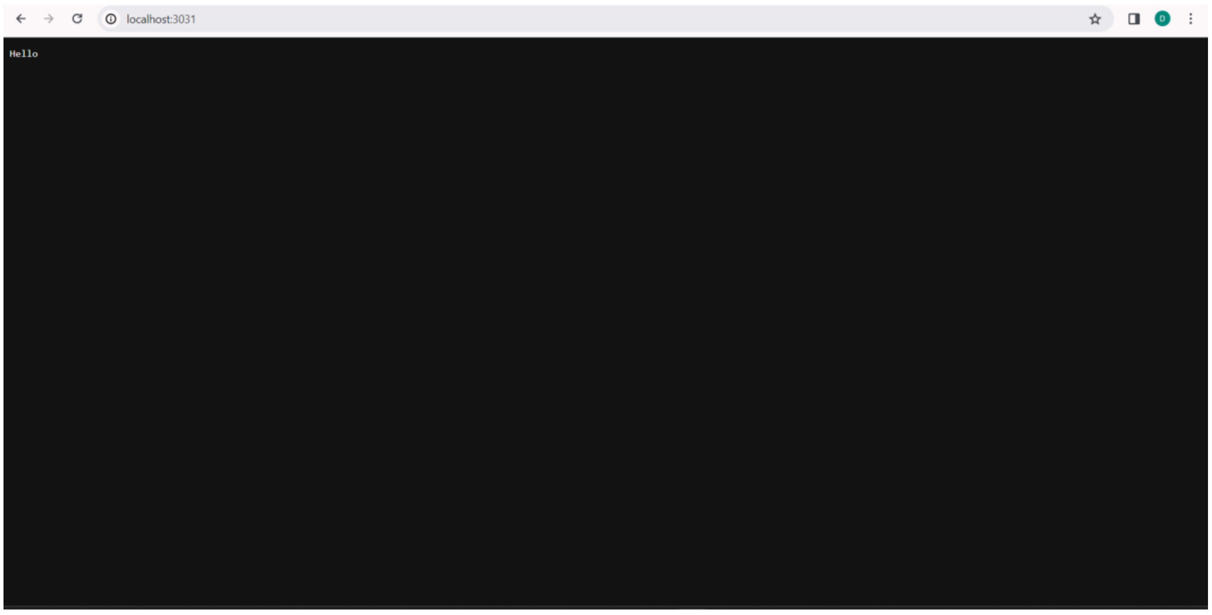
Code :

```
var http = require("http");
var server = http.createServer(function(req,res){
   res.writeHead(200,{'content.Type':'text/plain'});
   res.end("Hello");
});
server.listen(3031);
console.log("Server is running 127.0.0.1");
```

Output :



```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node exp-11.js
Server is running 127.0.0.1
```



localhost:3031

```
Hello
```

EXPERIMENT – 12:

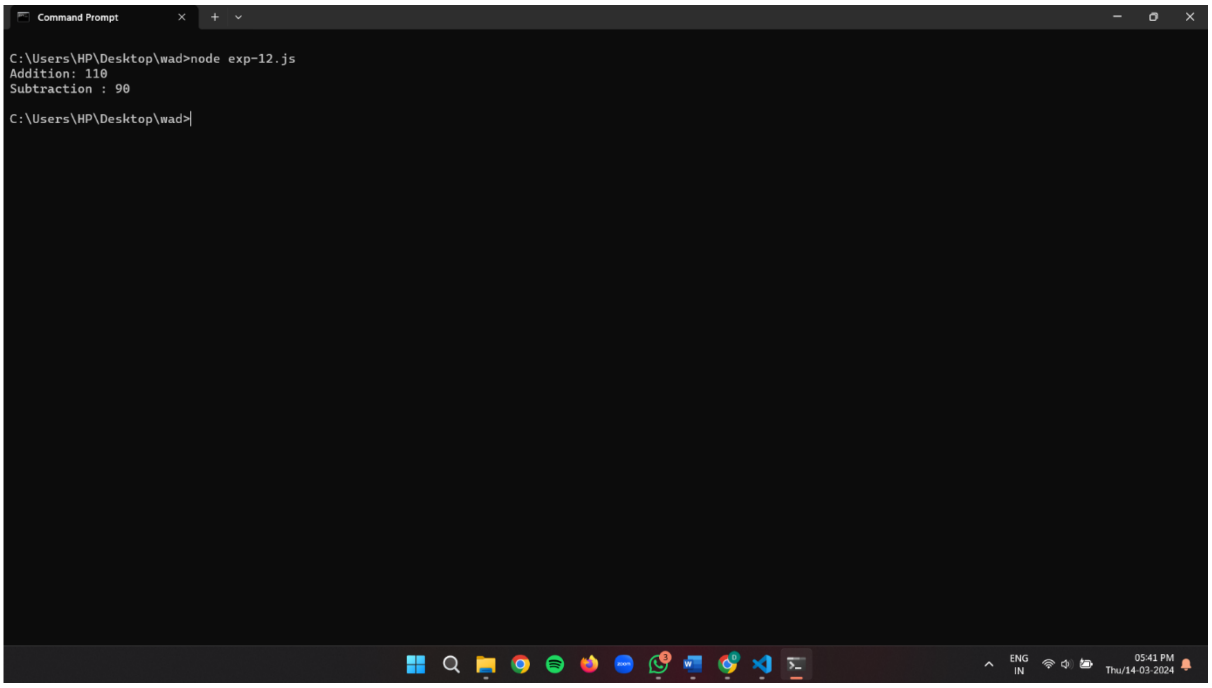
Demonstrate the process of creating and importing the user defined modules.

Code :

custom_module.js

```
exports.add_numbers = function(a,b){
   return a+b;
};
exports.subtract_numbers = function(a,b){
   return a-b;
};
var cm = require('./custom_module');
var a=100,b=10;
console.log("Addition: "+cm.add_numbers(a,b));
console.log("Subtraction : "+cm.subtract_numbers(a,b));
```

OUTPUT :



```
C:\Users\HP\Desktop\wad>node exp-12.js
Addition: 110
Subtraction : 90

C:\Users\HP\Desktop\wad>
```

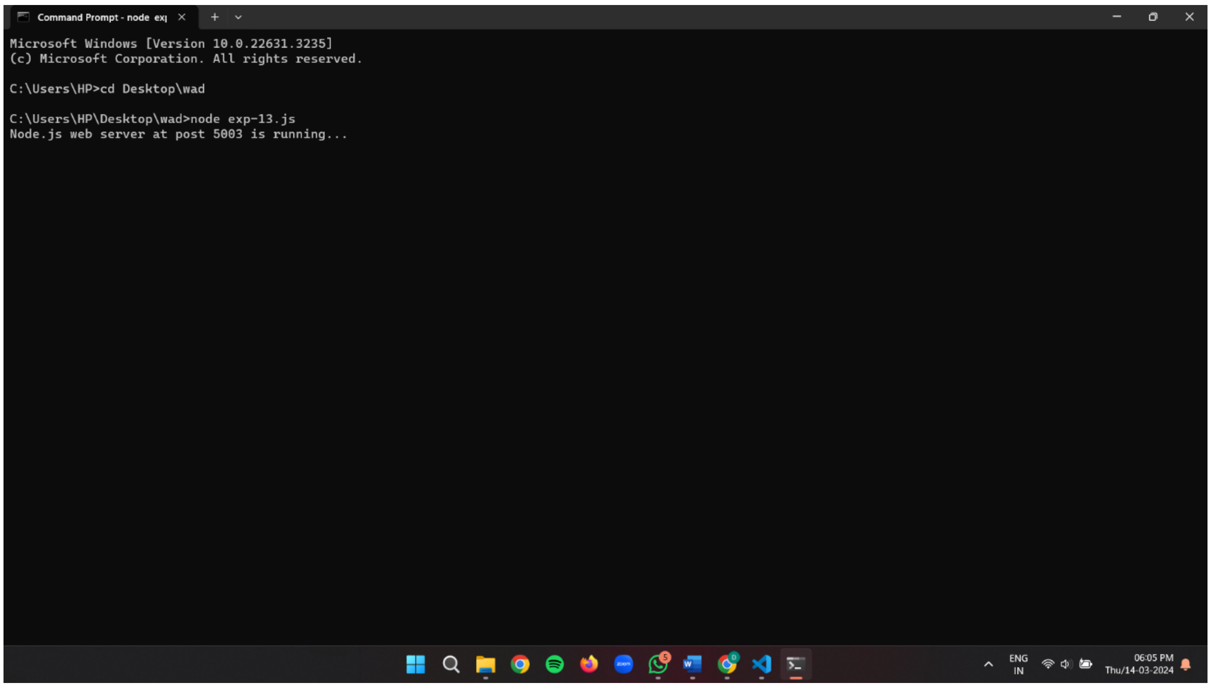
EXPERIMENT – 13

Demonstrate the process of creating web server and handling HTTP requests.

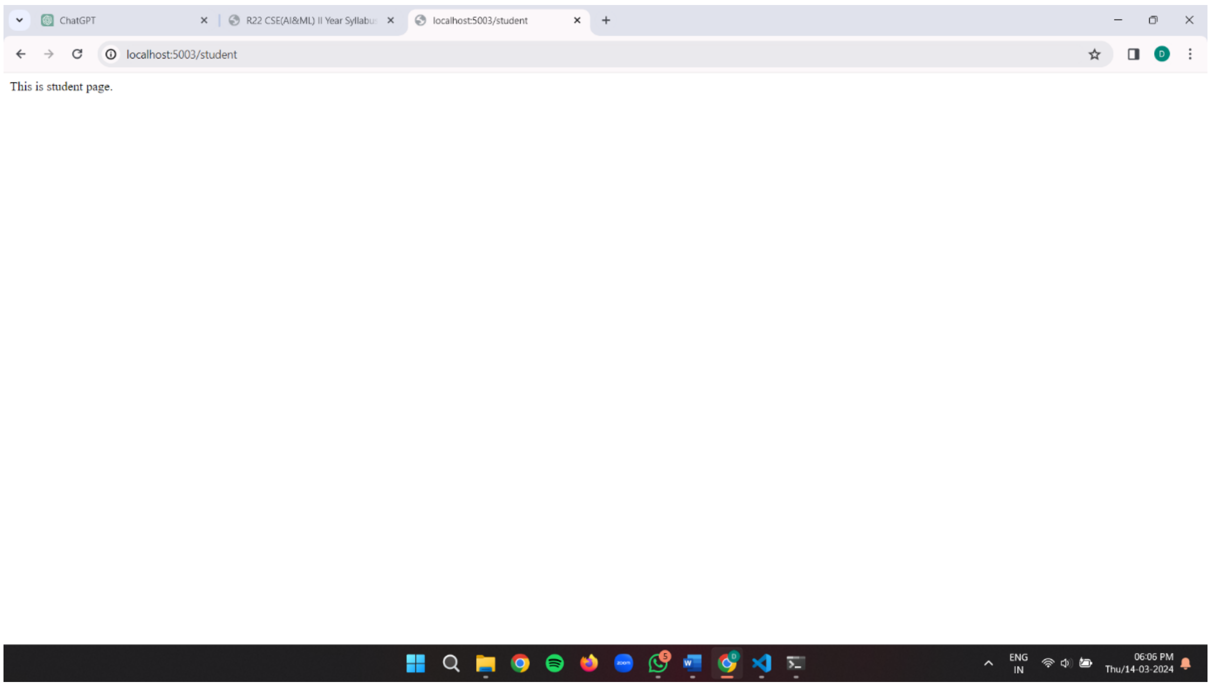
Code :

```
var http = require("http");
var server = http.createServer(function(req,res){
   if(req.url=='/'){
      res.writeHead(200,{'Content-Type':'text/html'});
      res.write('<html><body><p>This is home page.</p></body></html>');
      res.end();
   }
   else if(req.url=="/student"){
      res.writeHead(200,{'Content-Type':'text/html'});
      res.write('<html><body><p>This is student page.</p></body></html>');
      res.end();
   }
   else if(req.url=="/admin"){
      res.writeHead(200,{'Content-Type':'text/html'});
      res.write('<html><body><p>This is admin page.</p></body></html>');
      res.end();
   }
   else{
      res.end('Invalid response');
   }
});
server.listen(5003);
console.log("Node.js web server at post 5003 is running...")
```

OUTPUT :



```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node exp-13.js
Node.js web server at post 5003 is running...
```



localhost:5003/student

This is student page.

EXPERIMENT – 14:

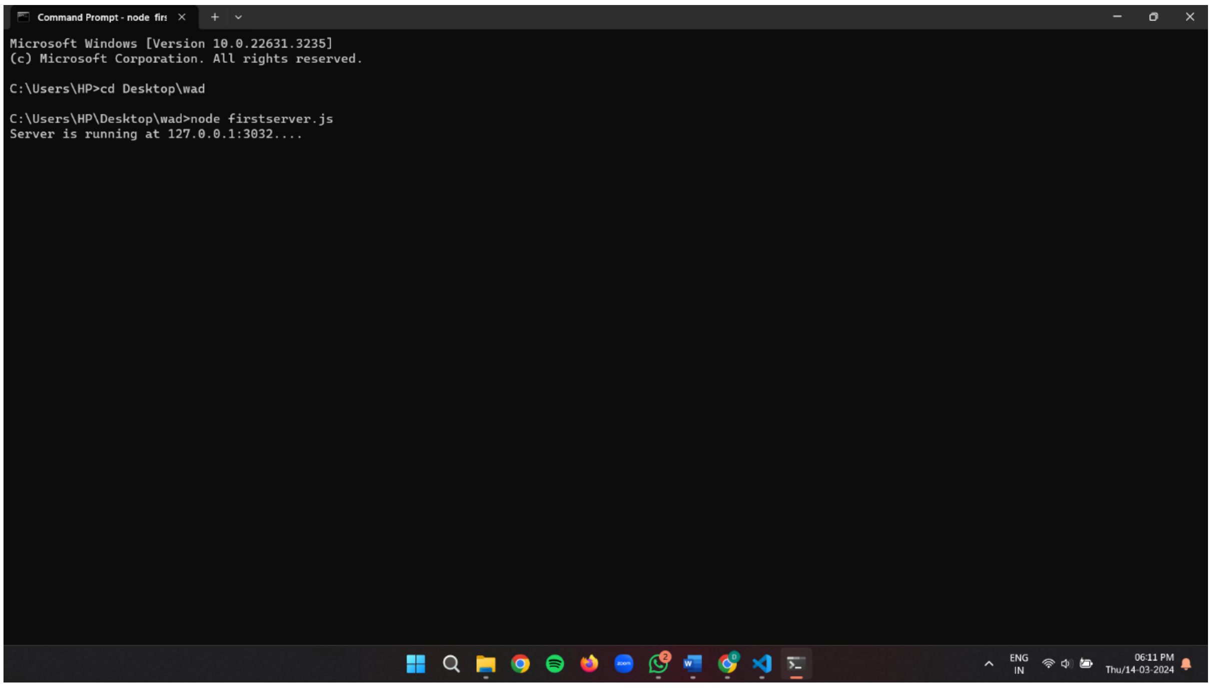
Illustrate the process of handling HTTP GET and POST request parameters and sending response to browser.

Code :

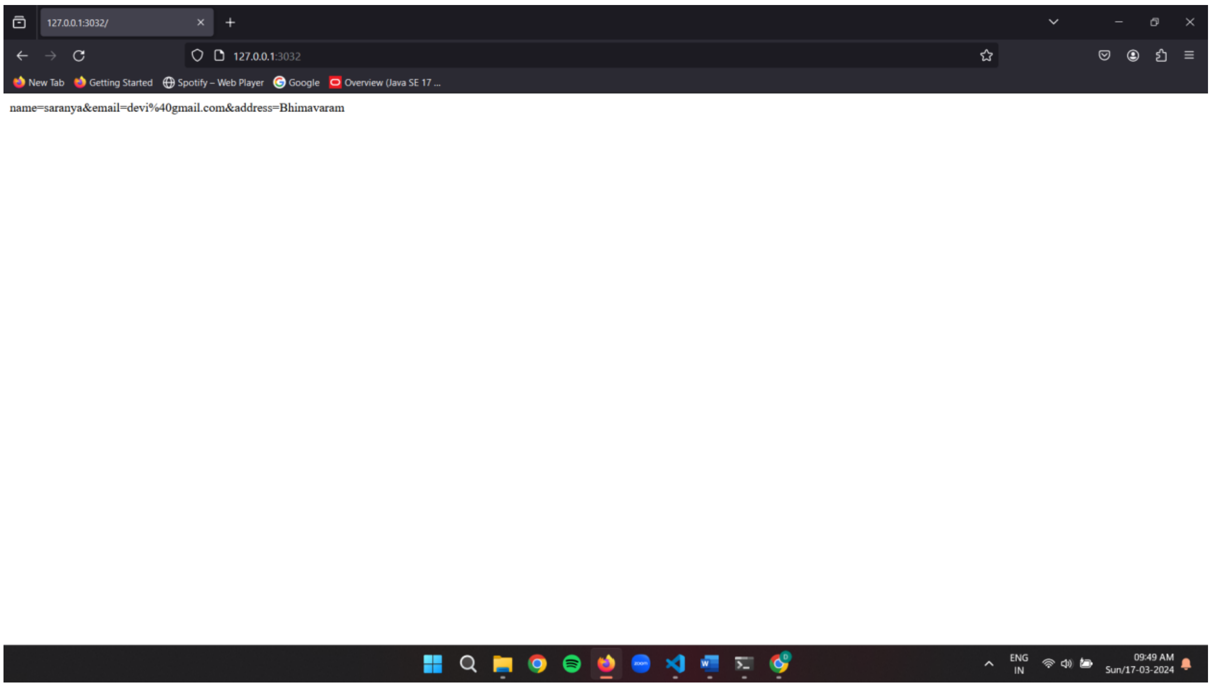
```
var http = require("http");

var url = require("url");

let server = http.createServer(function(req,res){

  if(req.method=="GET"){

      res.writeHead(200,{"Content-Type":"text/html"});

      var requrl=url.parse(req.url,true).query;


      var name = requrl.name;

      var email = requrl.email;

      var address = requrl.adddress;

      // const body = req.body;

      console.log(address);

      console.log(email);

      console.log(name);

      //fs.createReadStream("./ex.html","UTF-8").pipe(res);

      res.write("<html><body><p>"+name+" "+email+" "+address+"</p></body></html>");

      res.end();

  }

  else if(req.method=="POST"){

      var body = "";

      req.on("data",function(chunk){

          body += chunk;

      });

      req.on("end",function(){

          res.writeHead(200,{"Content-Type":"text/html"});

          res.end(body);

      });
```

```
   }


}).listen(3032);

console.log("Server is running at 127.0.0.1:3032....");
```

OUTPUT :



```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node firstserver.js
Server is running at 127.0.0.1:3032....
```



```
name=saranya&email=devi%40gmail.com&address=Bhimavaram
```

EXPERIMENT – 15:

Demonstrate the process of handling dynamic routes

Code :

Dynamic_routes.html

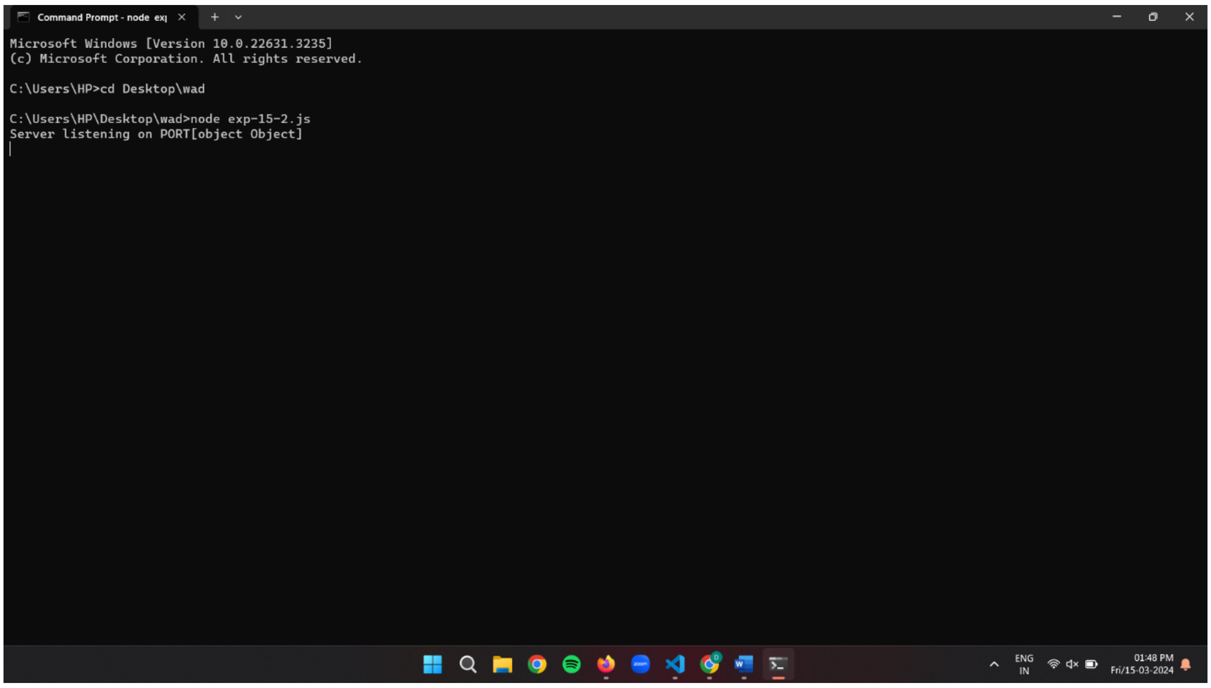
```
<!DOCTYPE html>
<head>
</head>
<body>
<form action="http://127.0.0.1:8000/" method="GET">
<input type="submit" value="BasicRoute">
</form>
<form action="http://127.0.0.1:8000/student/" method="GET">
<label>Name:</label>
<input type="text" name="dname" ><br>
<input type="submit" value="StudentRoute">
</form>
<form action="http://127.0.0.1:8000/admin/" method="GET">
<label>Department:</label>
<input type="text" name="department"><br>
<input type="submit" value="RouteDepartment">
</form>
</body>
</html>
```

Server.js

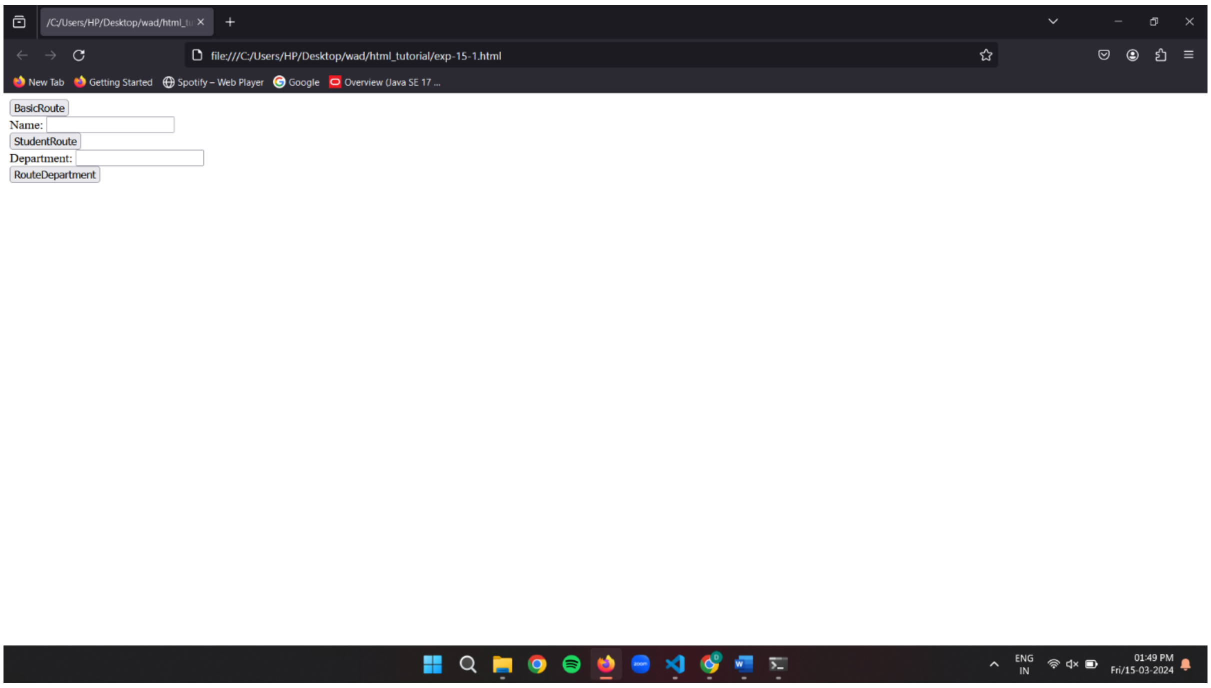
```
const express = require('express');
const url=require('url');
const app = express();
const PORT = 8000;
// Define routes as key-value pairs
const routes = {
```

```
'/':function(req,res){
res.send("Hello Wake Up World!");
},
   '/:student': function (req, res) {
//const name = req.params['dname'];
res.writeHead(200,{"Content-Type":"text/html"});
      var requrl=url.parse(req.url,true).query;
var name=requrl.dname;
res.write("<p>Name is:"+name+"</p>");
res.end();
   },
   '/:admin': function (req, res) {
      // let department = req.params;
res.writeHead(200,{"Content-Type":"text/html"});
      var requrl=url.parse(req.url,true).query;
//var department=requrl.department;
res.write("<p>Department is:"+requrl.department+"</p>");
res.end();
   },
};
// Loop through the routes and register
// them with Express
for (let route in routes) {
   app.get(route, routes[route]);
}
// Start the server
app.listen(PORT, function (err) {
   if (err) console.log(err);
   console.log("Server listening on PORT"+ {PORT});
});
```

OUTPUT :

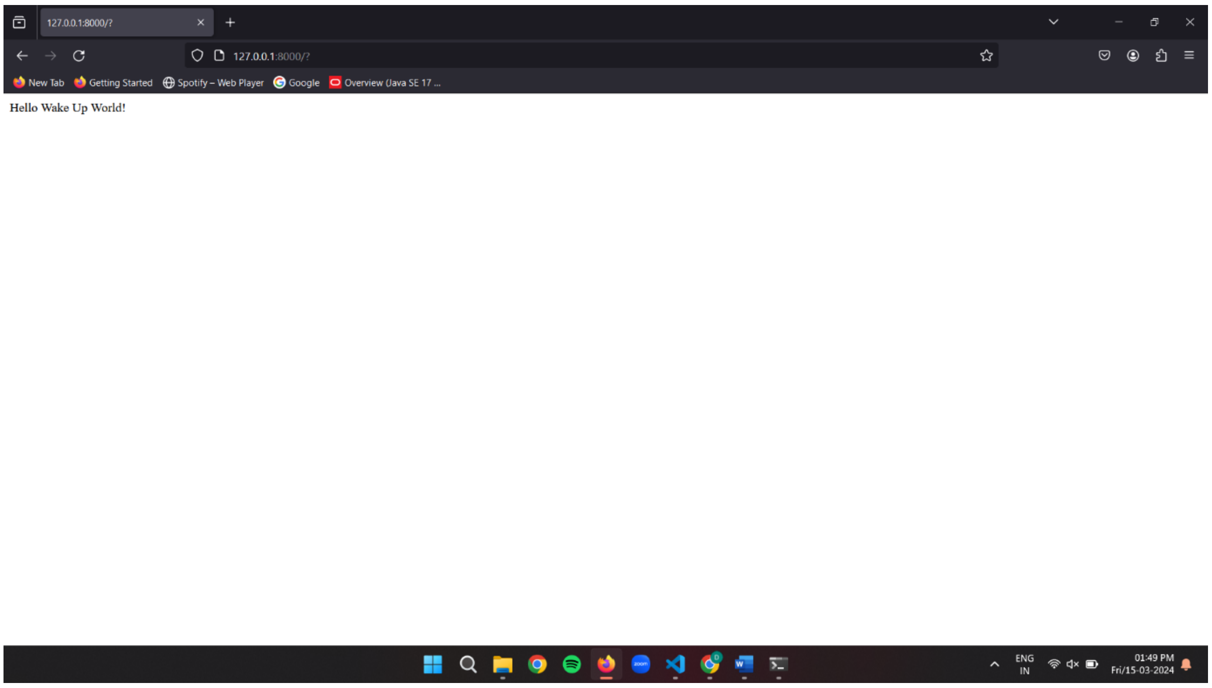


```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node exp-15-2.js
Server listening on PORT[object Object]
```

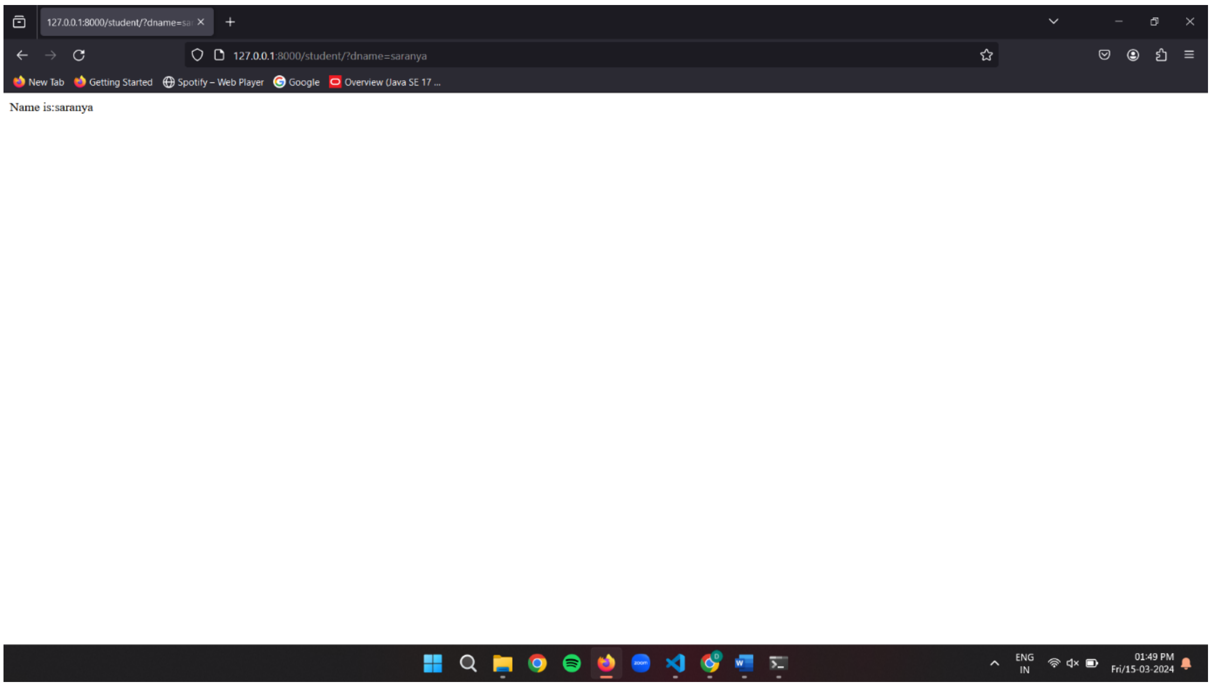


file:///C:/Users/HP/Desktop/wad/html_tutorial/exp-15-1.html

```
[BasicRoute]
Name: [          ]
[StudentRoute]
Department: [          ]
[RouteDepartment]
```



```
Hello Wake Up World!
```



```
Name is:saranya
```

EXPERIMENT – 16:

Demonstrate the file handling in NODE JS.

Code :

readme.txt

```
some text as
shri vishnu engineering college for women, vishnupur
```

readfile.js

```
var fs=require("fs");
var data=fs.readFileSync('readme.txt','utf-8');
console.log(data);
```

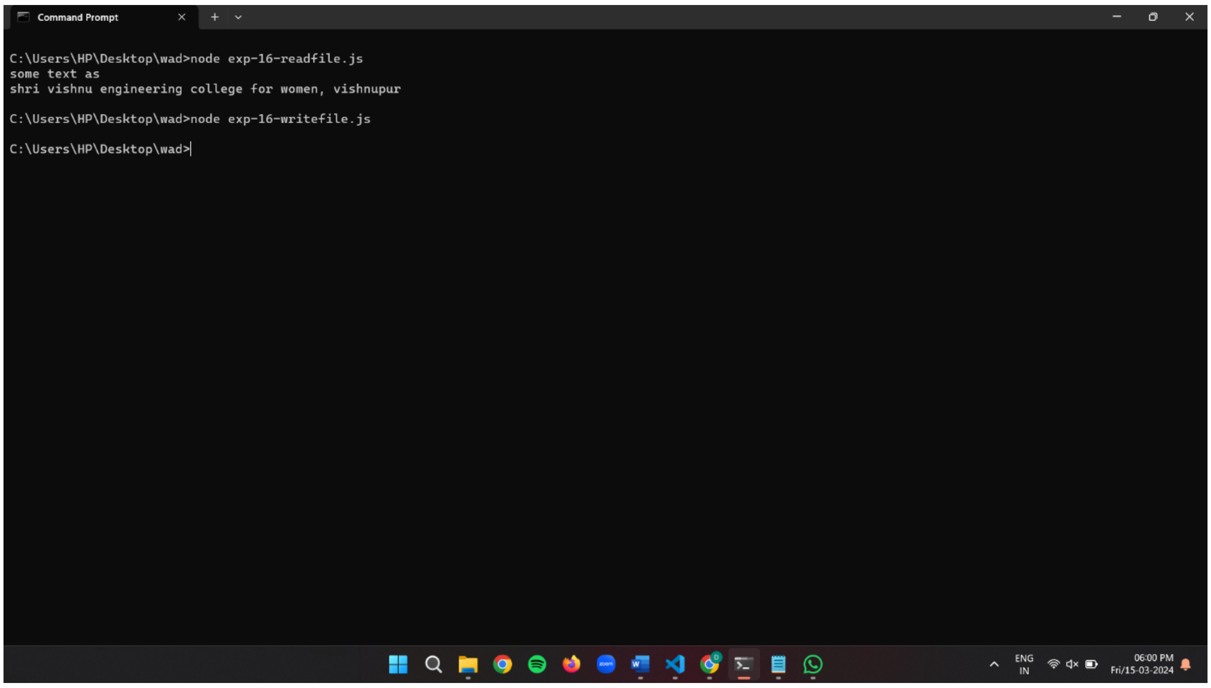
writefile.js

```
var fs=require("fs");
var data=fs.readFileSync('readme.txt','utf-8');
fs.writeFileSync('writemesync.txt',data);
```

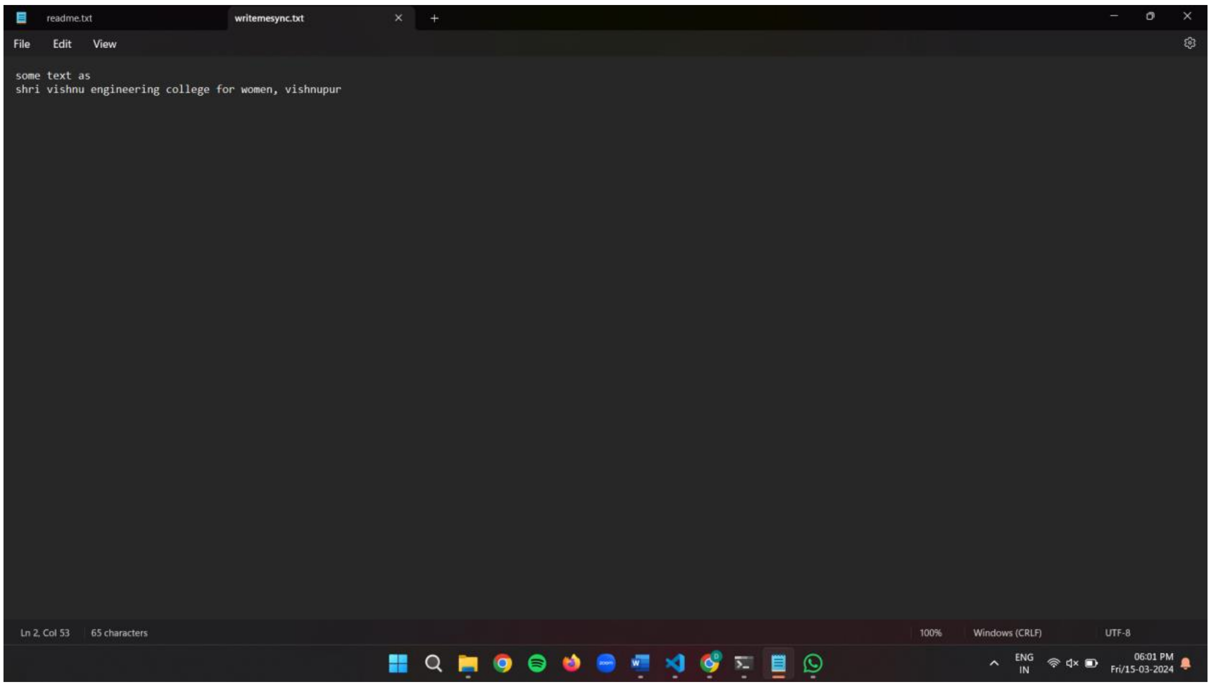
readwritefile.js

```
var fs=require("fs");
fs.readFile('readme.txt',function(err,data){
if(!err){
fs.writeFile('writeme.txt',data,(err)=>{
if(err)
throw err;
});
}
else
throw err;
});
```

OUTPUT :



```
C:\Users\HP\Desktop\wad>node exp-16-readfile.js
some text as
shri vishnu engineering college for women, vishnupur

C:\Users\HP\Desktop\wad>node exp-16-writefile.js

C:\Users\HP\Desktop\wad>
```



```
some text as
shri vishnu engineering college for women, vishnupur
```

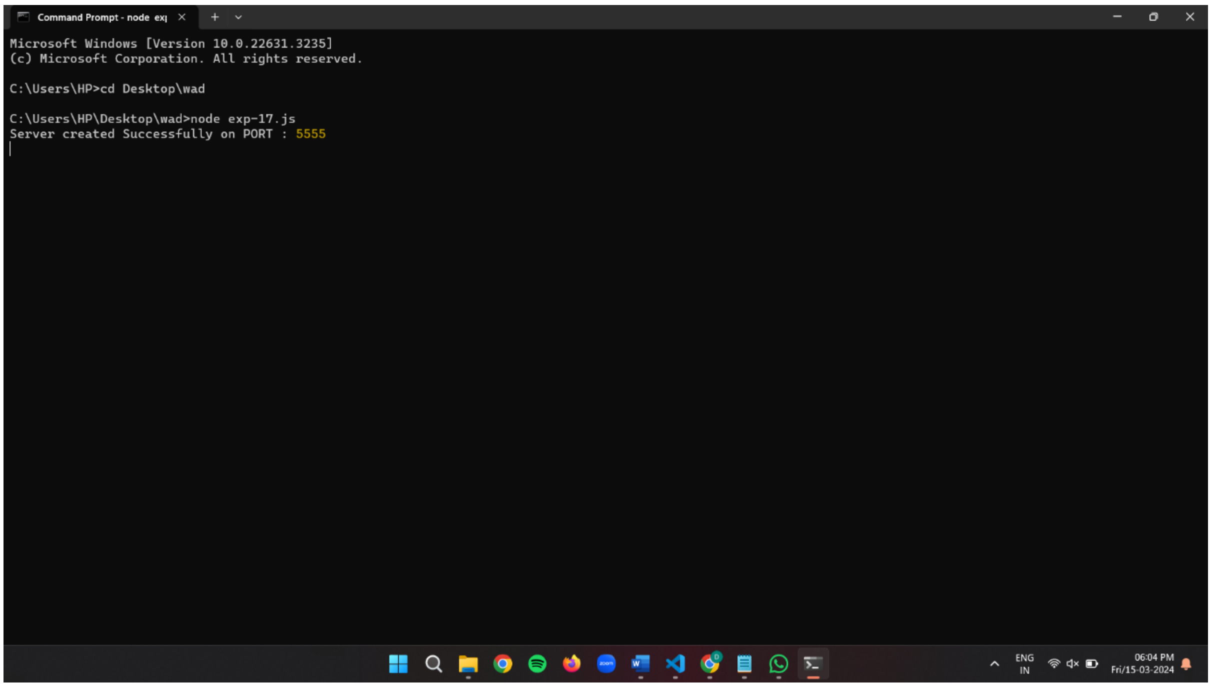
EXPERIMENT – 17:

Demonstrate how Session management takes place between several HTTP requests using express-session module.

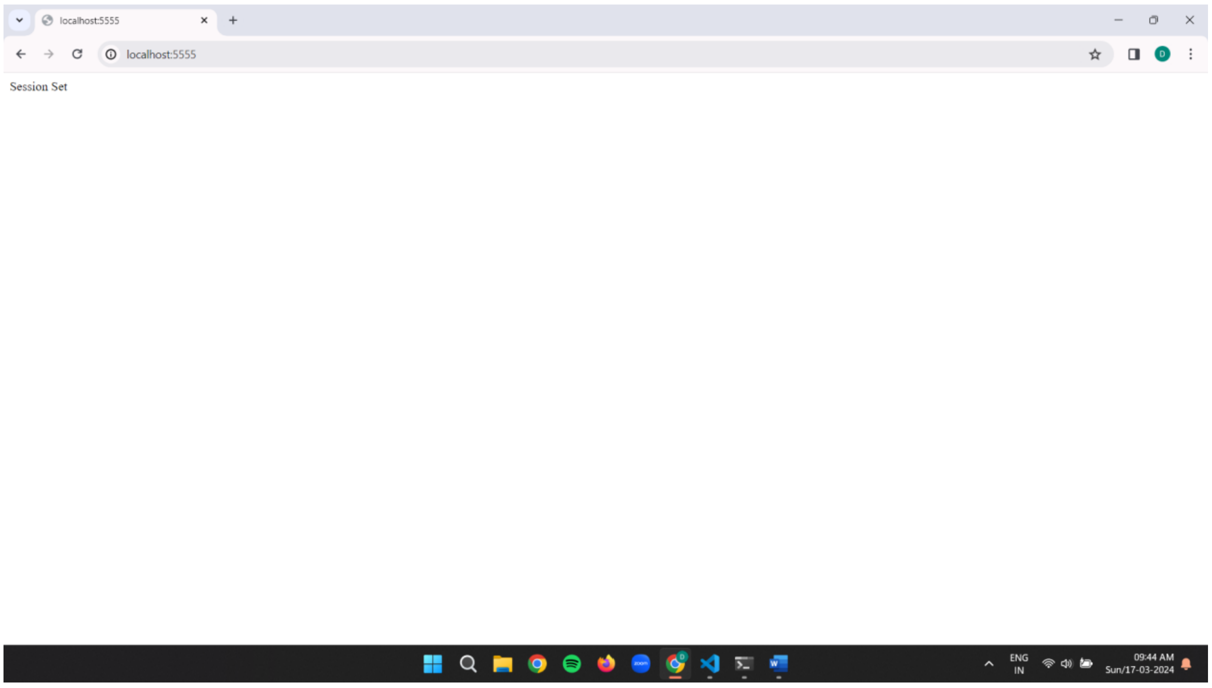
Code :

```
const express = require("express") ;
const session = require('express-session');
const app = express() ;
var PORT = 5555;
app.use(session({
   secret: 'svecw',
   resave: true,
   saveUninitialized: true
})) ;
app.get("/", function(req, res){
   req.session.name = 'Hello SVECW!';
   return res.send("Session Set") ;
}) ;
app.get("/session", function(req, res){
   var name = req.session.name ;
   return res.send(name)
   /*  To destroy session you can use
       this function
    req.session.destroy(function(error){
       console.log("Session Destroyed") ;
   })
   */
}) ;
app.listen(PORT, function(error){
   if(error) throw error
   console.log("Server created Successfully on PORT :", PORT)
}) ;
```

OUTPUT :



```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node exp-17.js
Server created Successfully on PORT : 5555
```



localhost:5555

```
Session Set
```

EXPERIMENT – 18

Demonstrate how to perform File upload and download from browser.

Code :

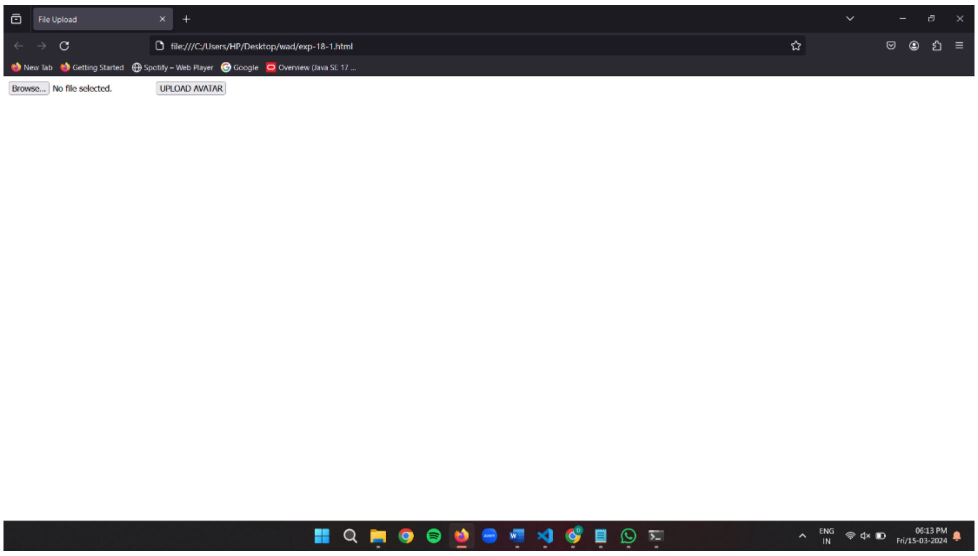
Index.html

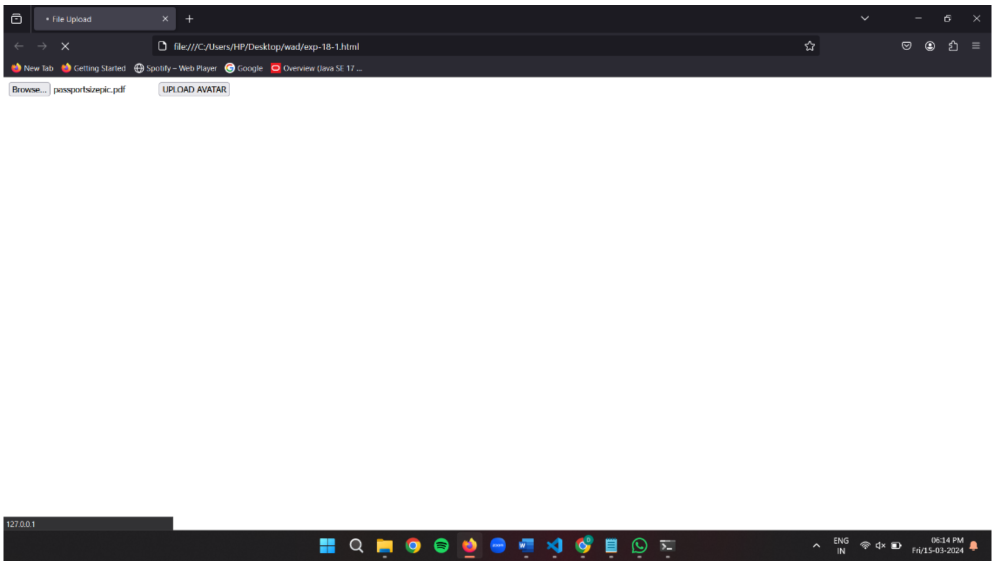
```
<!DOCTYPE html>
<html>
<head><title>File Upload</title>
</head>
<body>
<form action="http://127.0.0.1:8000/profile" method="post" enctype="multipart/form-data">
  <input type="file" name="avatar" />
  <input type="submit" value="UPLOAD AVATAR">
</form>
</body>
</html>
```

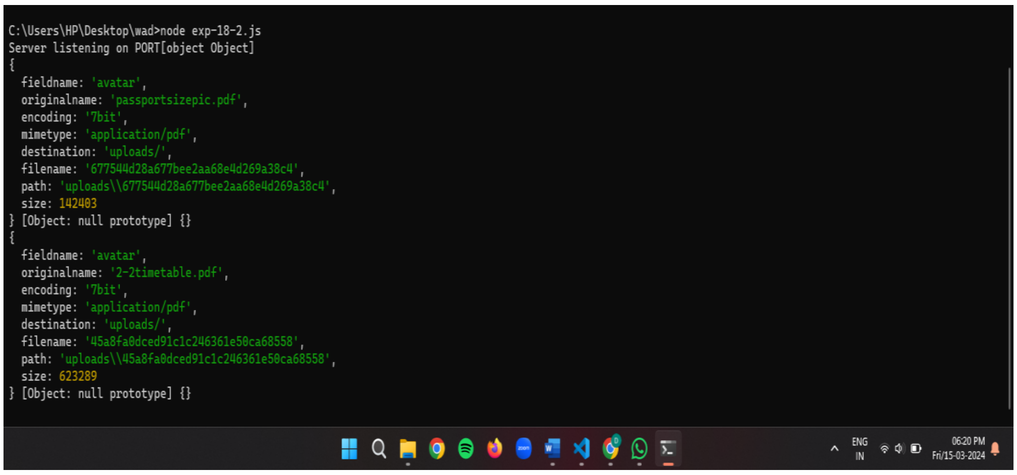
Index.js

```
const express = require('express');
const multer  = require('multer');
const upload = multer({ dest: 'uploads/' });
const url = require("url");
const PORT = 8000;
const app = express();
app.post('/profile', upload.single('avatar'), function (req, res, next) {
  console.log(req.file, req.body);
});
app.listen(PORT, function (err) {
   if (err) console.log(err);
   console.log("Server listening on PORT"+ {PORT});
});
```

OUTPUT :


File Upload
file:///C:/Users/HP/Desktop/wad/exp-18-1.html
Browse... No file selected.
UPLOAD AVATAR


File Upload
file:///C:/Users/HP/Desktop/wad/exp-18-1.html
Browse... passportsizepic.pdf
UPLOAD AVATAR
127.0.0.1



```
C:\Users\HP\Desktop\wad>node exp-18-2.js
Server listening on PORT[object Object]
{
  fieldname: 'avatar',
  originalname: 'passportsizepic.pdf',
  encoding: '7bit',
  mimetype: 'application/pdf',
  destination: 'uploads/',
  filename: '677544d28a677bee2aa68e4d269a38c4',
  path: 'uploads\\677544d28a677bee2aa68e4d269a38c4',
  size: 142403
} [Object: null prototype] {}
{
  fieldname: 'avatar',
  originalname: '2-2timetable.pdf',
  encoding: '7bit',
  mimetype: 'application/pdf',
  destination: 'uploads/',
  filename: '45a8fa0dced91c1c246361e50ca68558',
  path: 'uploads\\45a8fa0dced91c1c246361e50ca68558',
  size: 623289
} [Object: null prototype] {}
```

EXPERIMENT – 19:

Design server application with static HTML pages using Express module.

Code :

Userserver.js

```
var express=require('express')
var request = require('request')
var body=require('body-parser')
var app=express()
app.use(express.static('public'))
app.use(body.urlencoded({ extended: false }))
app.get('/',function(req,res) {
res.sendFile(__dirname+'/public/home.html')
})
app.get('/signup',function(req, res) {
res.sendFile(__dirname+'/public/signup.html');})
app.get('/login',function(req, res) {
 res.sendFile(__dirname+'/public/login.html');})
app.get('/logout',function(req,res){
res.send("this is logout page")})
app.listen(app.get('port'), function() {
console.log("Server is running");
})
```

Home.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Home</title>
   <link rel="stylesheet" type="text/css" href="bootstrap.css">
   <link rel="stylesheet" type="text/css" href="styles.css">
```

```
</head>

<body>

   <h1>SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN</h1>

   <TABLE>

      <tr>

   <td><a href="/signup"><button type="submit" class="btn btn-
success">SignUp</button></a>&nbsp;&nbsp;&nbsp;&nbsp;</td> <td><a href="/login"><button
type="submit" class="btn btn- primary">Login</button></a> &nbsp;&nbsp;&nbsp;&nbsp;</td>

</tr>

</TABLE>

</body>

</html>
```

Signup.html

```
<!DOCTYPE html>

<html>

<head>

<title>HTML Form</title>

<style type="text/css">

h2{

background-color: green; color: blue;

}

h3{

background-color: white; color: blue;

}

h3{

   background-color: #FFFFEE; color: blue;

}

</style>

</head>

<body style="background:red;color:white">
```

```
<form action="/loginSubmit" method="POST">
<table align="center">
<tr>
  <td>Name</td>
</tr>
<tr>
<td>
<input type="text" name="name">
</td>
</tr>
<tr>
<td>Last Name</td>
</tr>
<tr>
<td>
<input type="text" name="lname">
</td>
</tr>
<tr>
<td>E-Mail</td>
</tr>
<tr>
<td>
<input type="text" name="email">
</td>
</tr>
<tr>
<td>Mobile</td>
</tr>
<tr>
<td>
```

```
<input type="number" name="mobile">
</td>
</tr>
<tr>
<td>D.O.B</td>
</tr>
<tr>
<td>
<input type="date" name="dob">
</td>
</tr>
<tr>
<td>Year</td>
</tr>
<tr>
<td>
<select name="year">
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
</select>
</td>
</tr>
<tr>
   <td></td>
</tr>
<td> <input type="radio" name="g" value="Male">Male &nbsp;&nbsp;&nbsp;
<input type="radio" name="g" value="Female">Female
</td>
</tr>
```

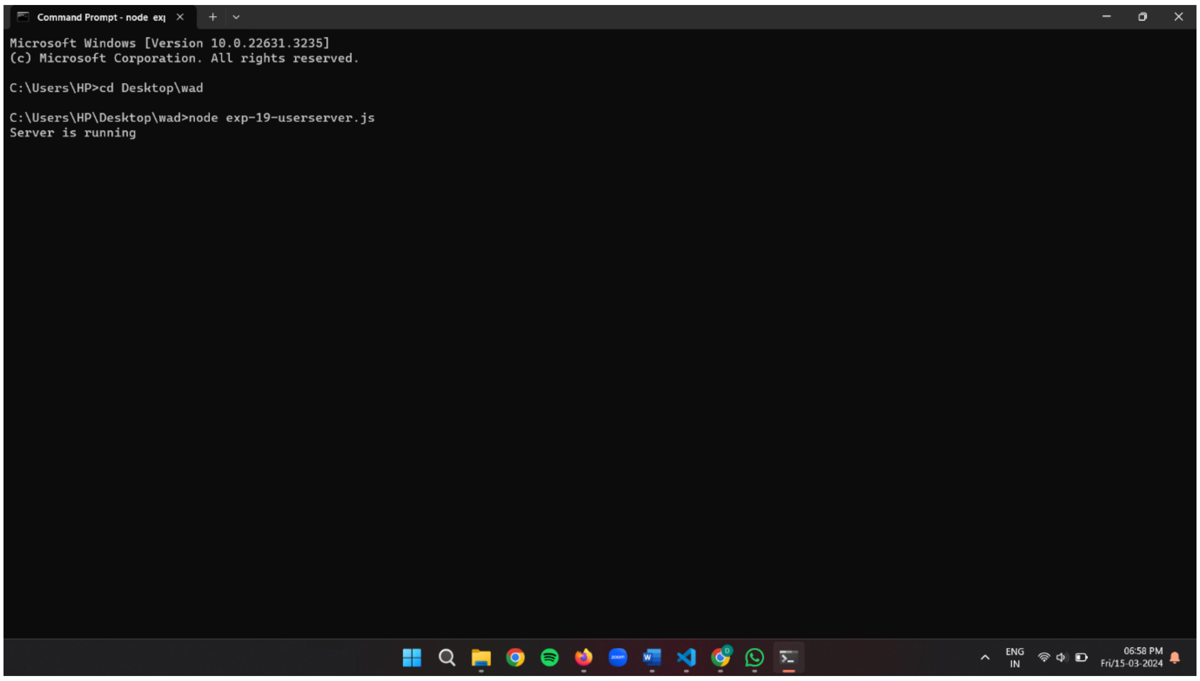
```
<tr>
<td></td>
</tr>
<tr>
<td>Do you like coding? <input type="checkbox" name="coding"value="Yes"></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>
<button type="Submit" class="btn btn-success">Submit</button>
</td>
</tr>
</table>
</form>
</body>
</html>
```

Login.html

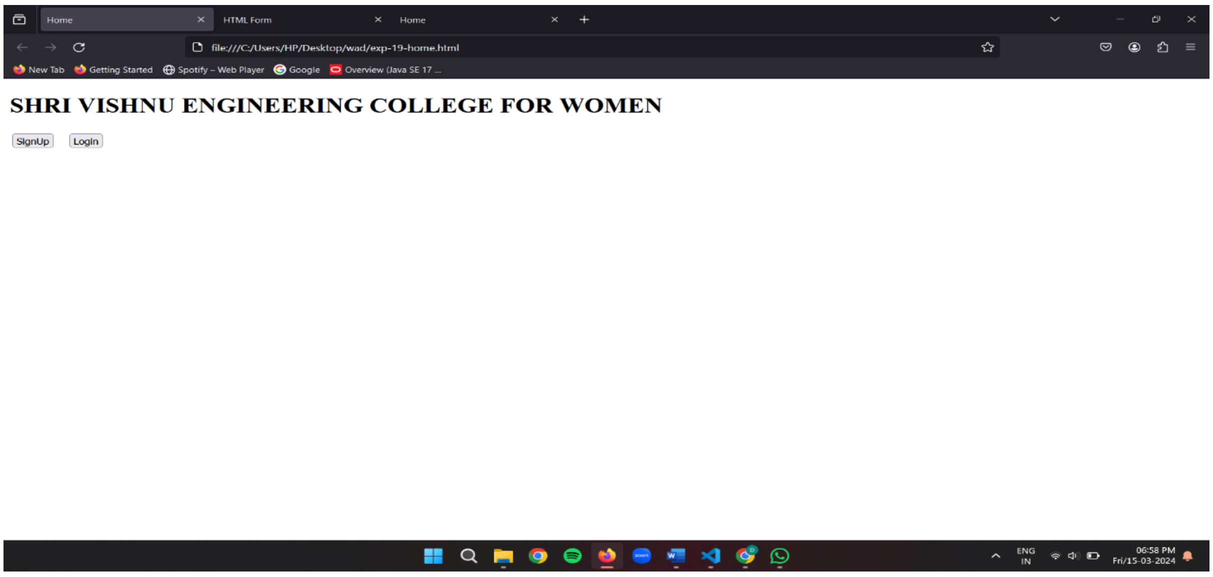
```
<!DOCTYPE html>
<html>
<head>
<title>Home</title>
<link rel="stylesheet" type="text/css" href="bootstrap.css">
<link rel="stylesheet" type="text/css" href="styles.css">
</head>
<h1>SHRI VISHNU ENGINERING COLLEGE FOR WOMEN</h1>
<body>
<form action="/loginpost" method="POST">
<center>
```

```
<TABLE>

   <tr>

   <td><input type=text name="uname" placeholder="User name"></td>

</tr>

<tr>

   <td><input type="password" name="password"placeholder="password"></td>

</tr>

<td colspan="2" align="center"><button type="Submit" class="btn btn-
primary">Submit</button></a> &nbsp;&nbsp;&nbsp;&nbsp;

</tr>

</TABLE>

</center>

</body>

</html>
```

OUTPUT :

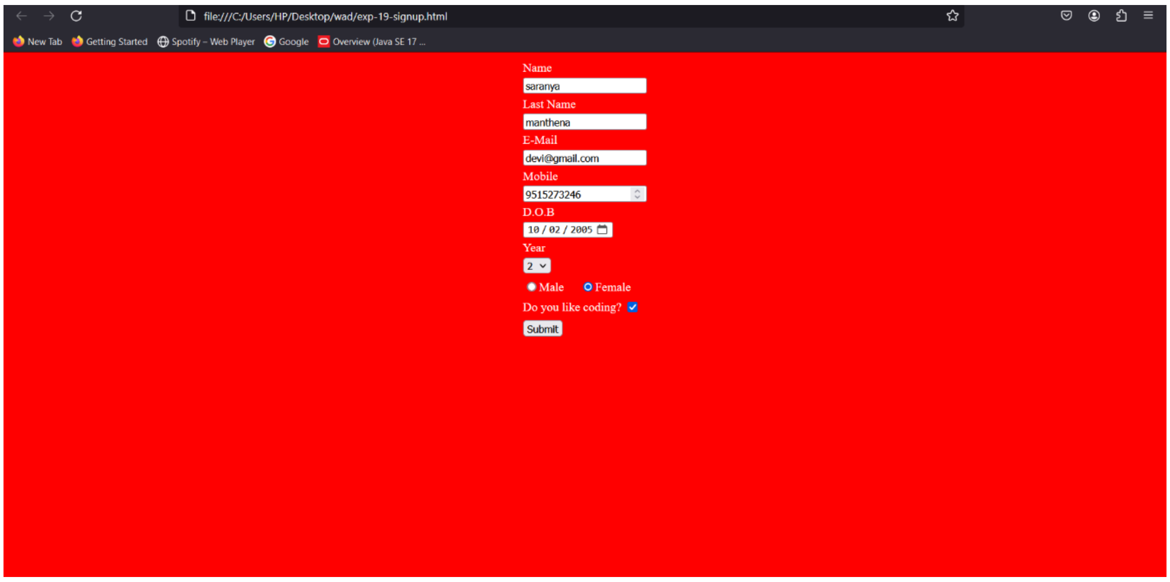


```
Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Desktop\wad

C:\Users\HP\Desktop\wad>node exp-19-userserver.js
Server is running
```

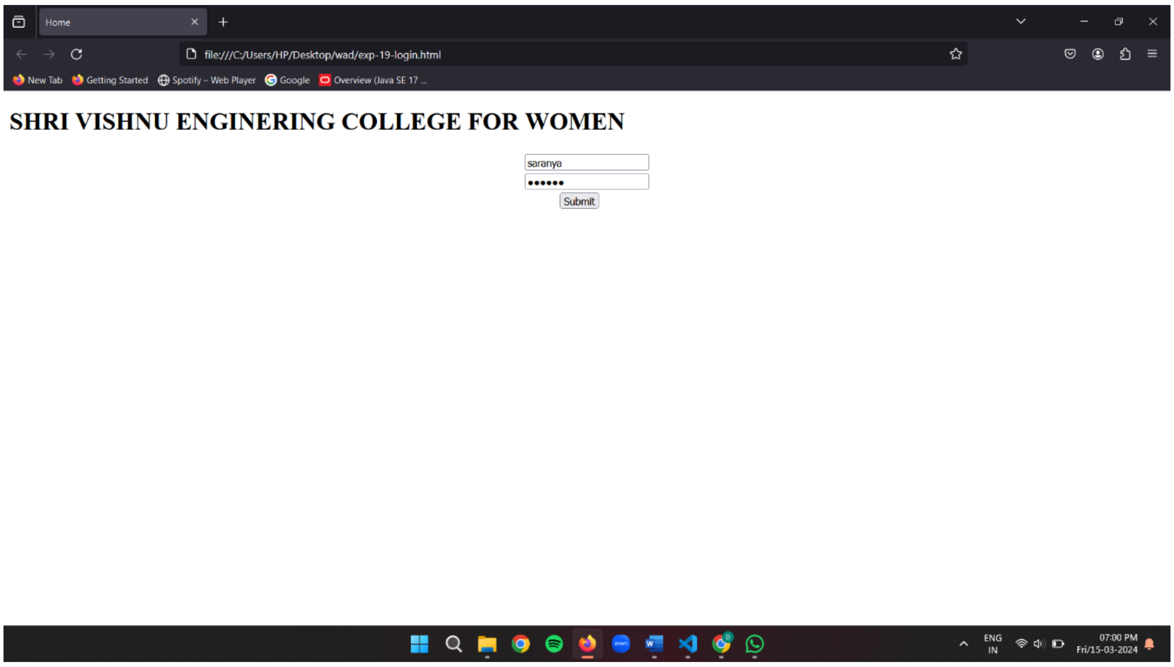


file:///C:/Users/HP/Desktop/wad/exp-19-home.html

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN


file:///C:/Users/HP/Desktop/wad/exp-19-signup.html

Name
saranya
Last Name
manthena
E-Mail
devi@gmail.com
Mobile
9515273246
D.O.B
10 / 02 / 2005
Year
2
Male [✓ Female]
Do you like coding? ✓
Submit


file:///C:/Users/HP/Desktop/wad/exp-19-login.html

SHRI VISHNU ENGINERING COLLEGE FOR WOMEN

saranya
••••••
Submit

07:00 PM
Fri/15-03-2024

EXPERIMENT – 20:

Design dynamic website using EJS (Embedded JavaScript Template) and Express.

Code :

Head.ejs

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

<title>EJS Is Fun</title>

<!-- CSS (load bootstrap from a CDN) -->

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.5.2/css/bootstrap.min.css">

<style>

  body { padding-top:50px; }

</style>
```

Header.ejs

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">

  <a class="navbar-brand" href="/">EJS Is Fun</a>

  <ul class="navbar-nav mr-auto">

    <li class="nav-item">

      <a class="nav-link" href="/">Home</a>

    </li>

    <li class="nav-item">

      <a class="nav-link" href="/about">About</a>

    </li>

  </ul>

</nav>
```

Footer.ejs

```
<p class="text-center text-muted">&copy; Copyright 2020 The Awesome People</p>
```

Index.js

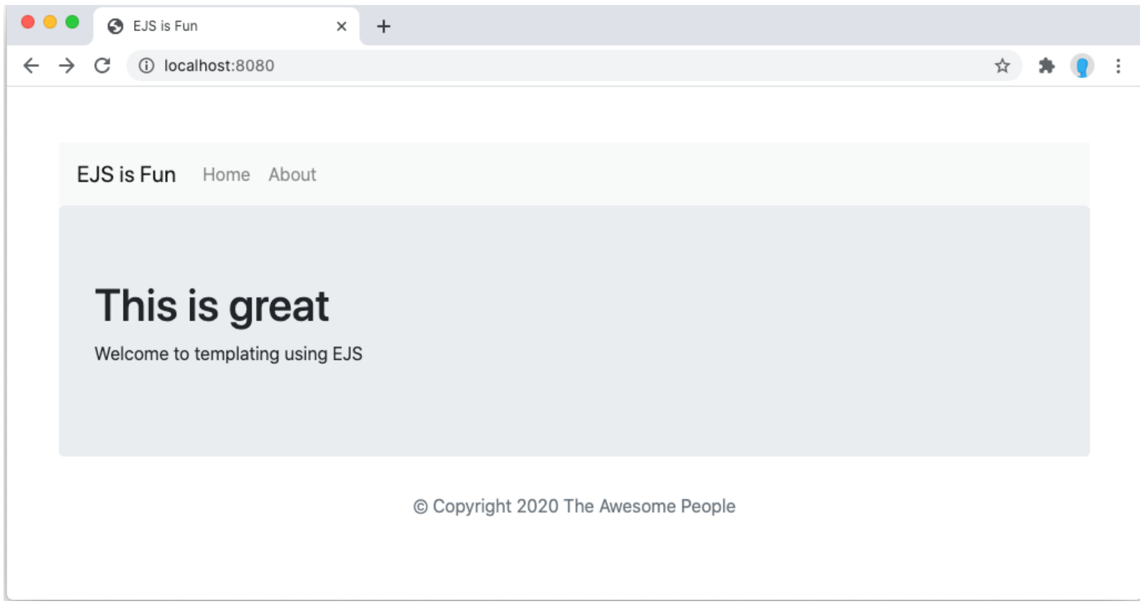
```
<!DOCTYPE html>

<html lang="en">

<head> <%- include('../partials/head'); %> </head>
```

```
<body class="container">
<header>
  <%- include('../partials/header'); %>
</header>
<main>
  <div class="jumbotron">
    <h1>This is great</h1>
    <p>Welcome to templating using EJS</p>
</div>
</main>
<footer>
  <%- include('../partials/footer'); %>
</footer>
</body>
</html>
```

Server.js

```
var express = require('express');
var app = express();
// set the view engine to ejs
app.set('view engine', 'ejs');
// use res.render to load up an ejs view file
// index page
app.get('/', function(req, res) {
  res.render('pages/index');
});
// about page
app.get('/about', function(req, res) {
  res.render('pages/about');
app.listen(8080); });
console.log('Server is listening on port 8080');
```

OUTPUT :



EJS is Fun Home About

This is great

Welcome to templating using EJS

© Copyright 2020 The Awesome People

EXPERIMENT – 21 :

Demonstrate the process of handling an API with sample application (Eg Show the top 100 movies from IMDB).

Code :

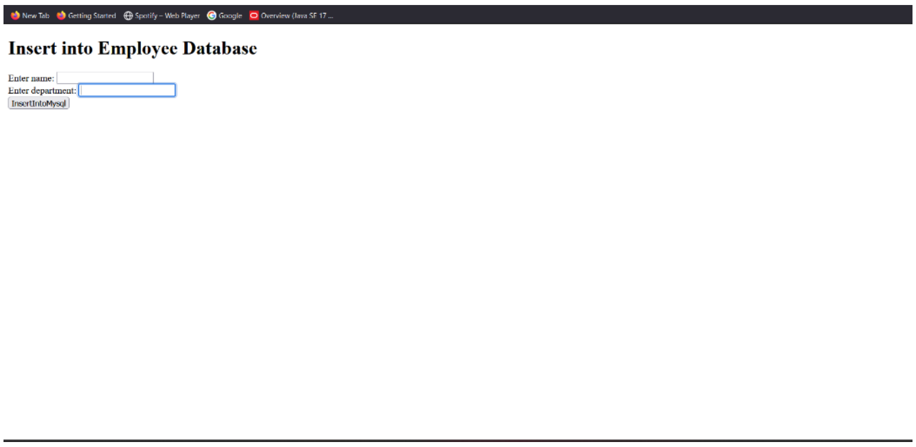
Employee.html

```
<!DOCTYPE html>
<html>
<head></head>
<body>
<h1>Insert into Employee Database</h1>
<form action="http://127.0.0.1:8000/insertintomysql/" >
<label>Enter name:</label>
<input type="text" name="dname">
<br>
<label>Enter department:</label>
<input type="text" name="department"><br>
<input type="submit" value="InsertIntoMysql">
</form>
</body>
</html>
```

InsertintomySQL.js

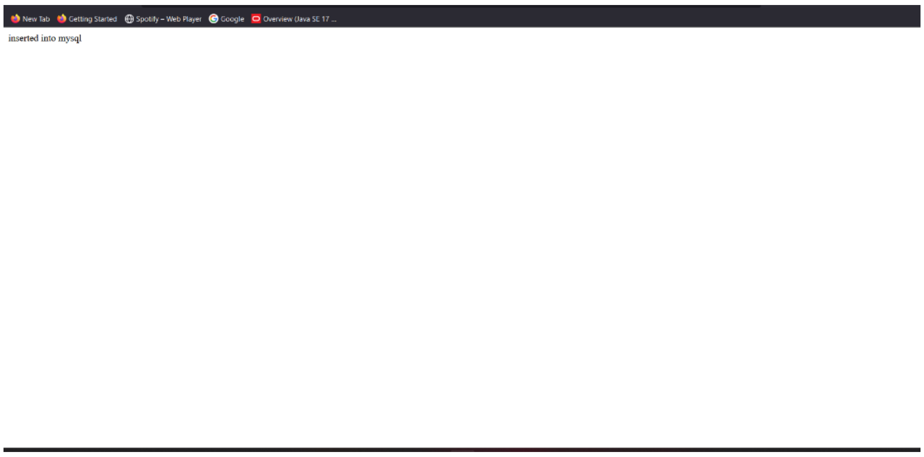
```
var http=require("http");
var sqlite3=require("sqlite3");
var url=require("url");
let server=http.createServer(function(req,res){
if (req.method=="GET"){
res.writeHead(200,{"Content-Type":"text/html"});
var requrl=url.parse(req.url,true).query;

var name=requrl.dname;
var department=requrl.department;
console.log(department);
console.log(name);
let db = new sqlite3.Database('./database.db');

  db.run("insert into employee values('"+name+"','"+department+"')",function(err,result){
    if (err) throw err;
    console.log(result);
  });
res.write("<html><body><p>"+"inserted into mysql"+"</p></body></html>");
res.end();
}

else if(req.method=="POST"){
var body="";
req.on("data",function(chunk){
body+=chunk;
});
req.on("end",function(){
res.writeHead(200,{"Content-Type":"text/html"});
res.end(body);
});
}

}).listen(8000);
console.log("Server is running at 127.0.0.1:8000.....");
```

OUTPUT :

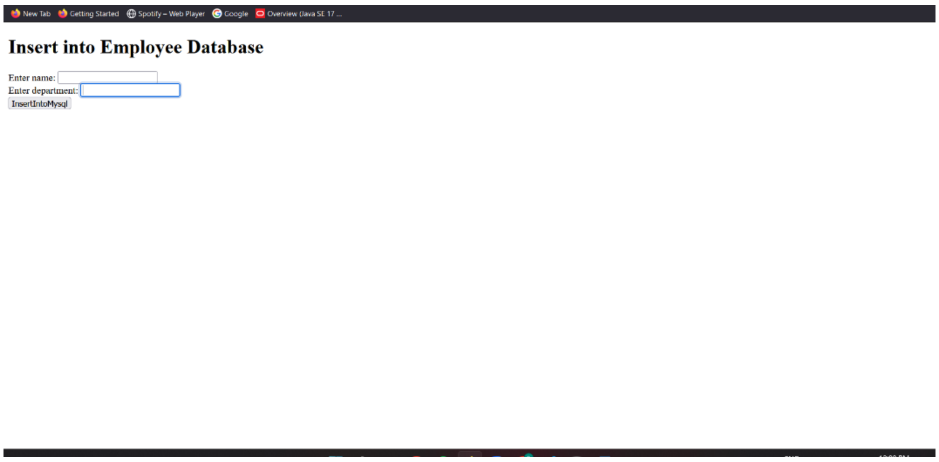

New Tab Getting Started Spotify – Web Player Google Overview (Java SE 17 ...

Insert into Employee Database

Enter name:
Enter department:
InsertIntoMysql


New Tab Getting Started Spotify – Web Player Google Overview (Java SE 17 ...

inserted into mysql


New Tab Getting Started Spotify – Web Player Google Overview (Java SE 17 ...

Insert into Employee Database

Enter name:
Enter department:
InsertIntoMysql

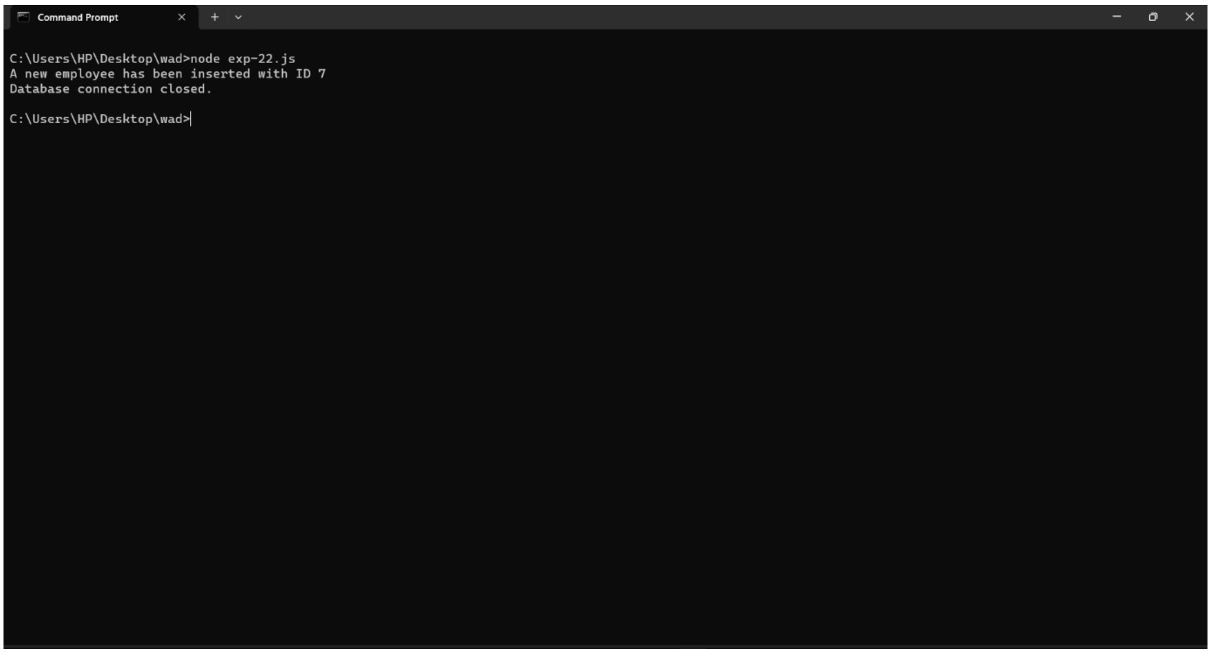
EXPERIMENT – 22:

Implement CRUD operations using SQL module.

Code :

```
const sqlite3 = require('sqlite3').verbose();
let db = new sqlite3.Database('./database.db');
let name = "John Doe";
let department = "Engineering";
db.run("INSERT INTO employee (name, department) VALUES (?, ?)", [name, department],
function(err) {
  if (err) {
    console.error(err.message);
  } else {
    console.log(`A new employee has been inserted with ID ${this.lastID}`);
  }
});
db.close((err) => {
  if (err) {
    console.error(err.message);
  } else {
    console.log('Database connection closed.');
  }
});
```

OUTPUT :



```
C:\Users\HP\Desktop\wad>node exp-22.js
A new employee has been inserted with ID 7
Database connection closed.

C:\Users\HP\Desktop\wad>
```

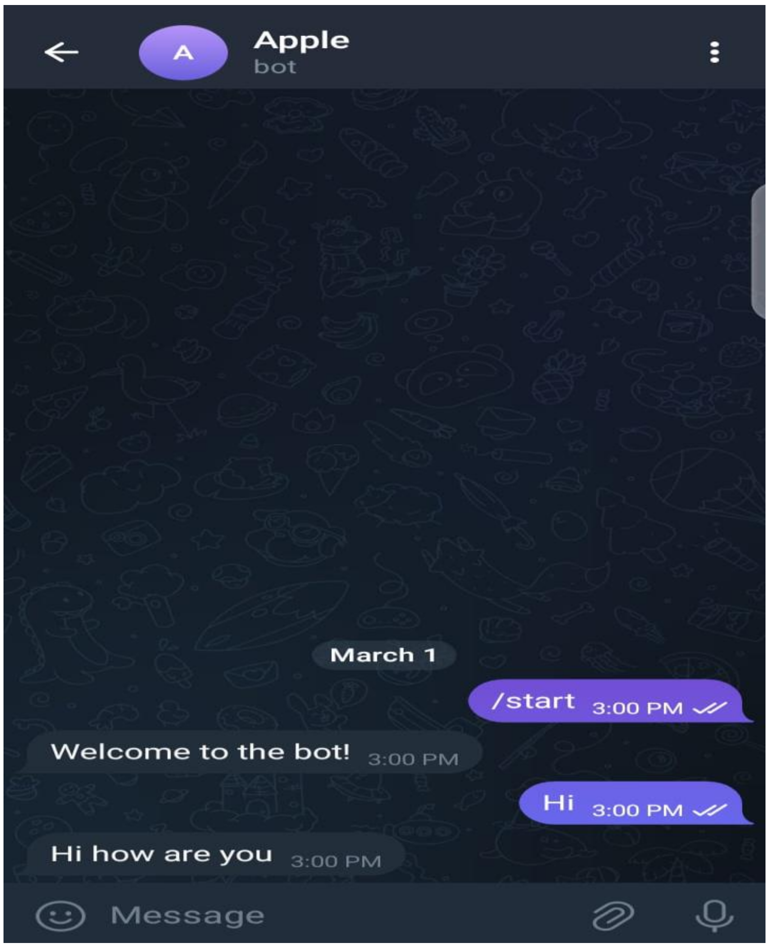
Experiment-23 :

Create Telegram ChatBot using telegram-bot-api module.

Code :

```
const TelegramBot = require('node-telegram-bot-api');
const token = '7009619409:AAEzFeWMkXl27mz3gNyQz3luvNqSSr8y_ak'
const bot = new TelegramBot(token, {polling: true});
bot.on('message',(msg) => {
    const chatId = msg.chat.id;
    const messageText = msg.text;
    if(messageText === '/start'){
        bot.sendMessage(chatId,'Welcome to the bot!');
    }
    else if(messageText === 'Hi') {
        bot.sendMessage(chatId,'Hi how are you')
    }
});
```

Output:



```
/start 3:00 PM
Welcome to the bot! 3:00 PM
Hi 3:00 PM
Hi how are you 3:00 PM
```