Practical:-1

Display your systems IP Address, Subnet mask using ipconfig, and find out the network address and the maximum number of systems possible on your network and range of IP addresses available to these systems.

Sol:- #Linux (ifconfig)

```
honeybee@xnactor:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 1500
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0xfe<compat,link,site,host>
        loop  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wifi0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.105  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::5e4f:d62:af7b:6c42  prefixlen 64  scopeid 0xfd<compat,link,site,host>
        ether 5c:ba:ef:91:66:db  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

#Windows (ipconfig)

```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::5e4f:d62:af7b:6c42%11
   IPv4 Address. . . . . . . . . . . : 192.168.3.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.3.1
```

IPv4 address – 192.168.3.105 (Class C)

Subnet Mask – 255.255.255.0

Broadcast address – 192.168.3.255

For Class C:

Total Networks: 2^21 = 2,097,152

Total Hosts for Each Network: 2^8 – 2 = 256 – 2 = 244

For This Network:

Network Address: (SubnetMask) AND (IP Address) = 192.168.3.0 Range of IP addresses for Hosts: 192.168.3.1 – 192.168.3.254

Practical2:-**With help of ping, check if you are connected to other systems of your network and find the route to connect to that system using tracert. List all the processes which are using ports for TCP protocol.**

Sol:- #Windows

List of Connected Devices

```
PS C:\Users\user> arp -a

  Internet Address      Physical Address      Type
  192.168.3.1           50-2b-73-7f-23-88     dynamic
  192.168.3.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.192.152.143       01-00-5e-40-98-8f     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

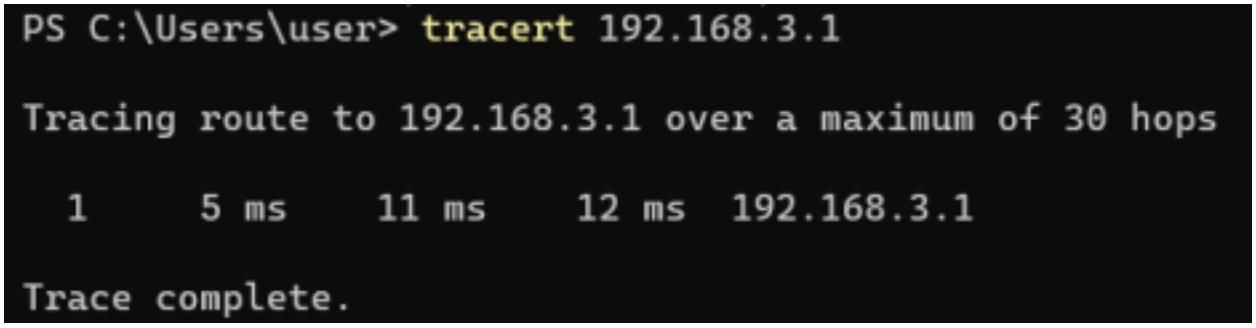
Pinging a device to check connection

```
PS C:\Users\user> ping 192.168.3.1

Pinging 192.168.3.1 with 32 bytes of data:
Reply from 192.168.3.1: bytes=32 time=14ms TTL=64
Reply from 192.168.3.1: bytes=32 time=6ms TTL=64
Reply from 192.168.3.1: bytes=32 time=13ms TTL=64
Reply from 192.168.3.1: bytes=32 time=28ms TTL=64

Ping statistics for 192.168.3.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 28ms, Average = 15ms
```

Tracing the route to the selected device



List of processes using ports for TCP protocol

```
Proto  Local Address          Foreign Address        State
TCP    127.0.0.1:49671        xnactor:49672          ESTABLISHED
TCP    127.0.0.1:49672        xnactor:49671          ESTABLISHED
TCP    127.0.0.1:49673        xnactor:49674          ESTABLISHED
TCP    127.0.0.1:49674        xnactor:49673          ESTABLISHED
TCP    192.168.3.105:60404    20.198.119.84:https    ESTABLISHED
TCP    192.168.3.105:60410    20.198.119.84:https    ESTABLISHED
TCP    192.168.3.105:60834    a23-49-71-11:https     CLOSE_WAIT
TCP    192.168.3.105:65047    a23-217-111-75:https   ESTABLISHED
TCP    192.168.3.105:65049    40.99.9.178:https      TIME_WAIT
TCP    192.168.3.105:65054    13.107.246.48:https    CLOSE_WAIT
TCP    192.168.3.105:65056    13.107.19.254:https    ESTABLISHED
TCP    192.168.3.105:65057    204.79.197.222:https   ESTABLISHED
TCP    192.168.3.105:65062    25:https               TIME_WAIT
TCP    192.168.3.105:65064    237:4070               TIME_WAIT
TCP    192.168.3.105:65066    25:https               TIME_WAIT
TCP    192.168.3.105:65067    13.107.246.68:https    CLOSE_WAIT
TCP    192.168.3.105:65068    146.75.118.248:https   ESTABLISHED
TCP    192.168.3.105:65070    25:https               TIME_WAIT
TCP    192.168.3.105:65075    47:https               TIME_WAIT
TCP    192.168.3.105:65076    204.79.197.254:https   ESTABLISHED
TCP    192.168.3.105:65077    249:https              TIME_WAIT
TCP    192.168.3.105:65082    del03s10-in-f2:https   TIME_WAIT
TCP    192.168.3.105:65083    13.107.6.254:https     ESTABLISHED
TCP    192.168.3.105:65084    13.107.237.254:https   ESTABLISHED
TCP    192.168.3.105:65085    del12s02-in-f2:https   TIME_WAIT
TCP    192.168.3.105:65086    152.199.43.62:https    ESTABLISHED
TCP    192.168.3.105:65095    del12s02-in-f2:https   TIME_WAIT
TCP    192.168.3.105:65096    del12s06-in-f2:https   TIME_WAIT
TCP    192.168.3.105:65097    del12s08-in-f1:https   TIME_WAIT
TCP    192.168.3.105:65098    del11s14-in-f1:https   TIME_WAIT
TCP    192.168.3.105:65100    del12s06-in-f2:https   TIME_WAIT
TCP    192.168.3.105:65103    loft10072:82           TIME_WAIT
TCP    192.168.3.105:65107    40.99.9.178:https      ESTABLISHED
TCP    192.168.3.105:65108    a23-217-111-66:https   ESTABLISHED
TCP    192.168.3.105:65110    47:https               TIME_WAIT
TCP    192.168.3.105:65112    237:4070               ESTABLISHED
TCP    192.168.3.105:65113    25:https               ESTABLISHED
TCP    192.168.3.105:65115    25:https               ESTABLISHED
```

Practical3:-

Create an HTML page that shows information about you, your course, hobbies, address, and your plans. Use CSS for styling of HTML page so that looks nice.

Sol:-

```
<!DOCTYPE html>
<html lang="eng">

<head>
    <title>Practical 3</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
    scale=1.0"> <style type="text/css">
        * {}

        body {
            background: grey;
        }

        .container {
            width: 30%;
            min-width: 330px;
            margin: auto;
            margin-top: 10px;
            margin-bottom: 10px;
            background-image: url('https://external
content.duckduckgo.com/iu/?=images');
            background-size: contain;
            border: 1px solid white;
            padding: 10px;
            border-radius: 8px;
        }

        .my-5 {
            margin-top: 5px;
            margin-bottom: 5px;
        }

        .underline {
            text-decoration: underline;
        }

        .main {
            display: flex;
            flex-wrap: wrap;
        }

        .sub-head {
            font-weight: bold;
        }
```

```
    </style>
</head>

<body>
    <div class="main">
        <div class="container">
            <h1 class="my-5 underline">Vikram</h1>
            <p>Bsc. Computer Science Hons.</p>
        </div>
        <div class="container">
            <p> <span class="underline sub-head">Hobbies:</span> <span>
Cricket,  Reading, Learing about new
                    things</span> </p>
            <p> <span class="underline sub-head">Address:</span> <span>
Narnaul,  Haryana</span> </p>
        </div>
        <div class="container">
            <p> <span class="underline sub-head">Plans:</span>
            <ul>
                <li>Learn a new language</li>
               <li>Stay fit and maintain good work life balance</li>
                <li>Be independent</li>
            </ul>
            </p>
        </div>
    </div>
</body>
</html>
```

Practical:-4

Create an HTML page with the sole purpose to show multiplication tables of 2 to 10 (row-wise) created by JavaScript. Initially, the page is blank. With help of setInterval function print a row every 5 seconds in different colors and increasing font size.

Sol:-
```
<!DOCTYPE html>
<html>
<head>
<title>Practical 4</title>
<meta charset="utf-8">
<style type="text/css">
table {
border: 1px solid black;
border-collapse: collapse;
width: 80%;
margin: auto;
```

```
}
td,
th {
border: 1px solid black;
padding: 5px;/border-collapse: collapse;/
}
.center {
text-align: center;
}
</style>
</head>
<body>
<h1>Printing Table from 2 to 10</h1>
<table class="center" id="content"> </table>
<script
type="text/javascript"> function getRandomColor() { var letters =
'0123456789ABCDEF'; var color = '#'; for (var i = 0; i < 6; i++) { color +=
letters[Math.floor(Math.random() * 16)]; } return color; } var tbl =
document.getElementById('content'); var number = 2; var abc =
setInterval(printrow, 5000); function printrow() {

if (number == 10) { clearInterval(abc); } var result = ""; for
(var i = 1; i <= 10; i++) { result = result + "<td>" + number + "*" + i + "=" +
number * i + "</td>"; } number++; var row = document.createElement('tr');
row.style.color = getRandomColor(); row.style.fontSize = (number + 10) + "px";
row.innerHTML = result; tbl.append(row);

} </script>

</body>
</html>
```

Practical:-5 Create an HTML page with a paragraph written on it and under which 9 buttons are placed in a 3X3 grid. The first row is for buttons labeled with colors names Red, Green, and Blue, the second row with numbers 10, 20, 30, and the third row with different font names. Click event of each of the buttons should make the appropriate change in the style of paragraph.

Sol:-
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Practical 6</title>
<style type="text/css">
.container {
width: 60%;
```

```
margin: auto;
border: 1px solid black;
border-radius: 8px;
padding: 50px;
}
.btn-submit {
border-radius: 5px;
color: white;
background: greenyellow;
font-weight: bold;
font-size: 1rem;
margin: 20px;
}
@media(width<=575) {
.container {
width: 84%;
}
}
</style>
</head>
<body>
<div class="container">
<h1>Pet's Information</h1>

<hr> <label for="name">Pet's Name:</label> <input type="text"
name="name"><br><br> <label for="age">Age:</label>
<input type="number" name="age"> <label for="weight">Weight:</label>
<input type="number" name="weight"

class=""><br><br> <label for="type">Pet type:</label> <input

type="text" name="type"><br><br>
<label for="likes">Likes:</label> <input type="text" name="likes"><br>
<button class="btn-submit"

onclick="display()">Submit</button>
</div>
<script
type="text/javascript"> function display() {
// event.preventDafault();
var pet = {};
var input_fields = document.getElementsByTagName('input');
for (var i =0; i < input_fields.length; i++) {

pet[input_fields[i].name] = input_fields[i].value; } console.log(pet); }
</script>
```

```
</body>
</html>
```

Practical:-6 Store JSON data of few pets that you created in previous practical in a JSON file (copy from console output of previous program to a .json file). Using AJAX, load data from the file and display it in a presentable way using HTML and CSS.

Sol:-

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Practical 7</title>
<style type="text/css">
#pet-data {
border: 1px solid black;
border-radius: 10px;
border-collapse: collapse;
}
td {
border: 1px solid black;
border-collapse: collapse;
}
#btn-fetch {
margin-top: 20px;
font-size: 24px;
font-weight: bold;
background-color: black;
color: white;
border-radius: 8px;
}
</style>
</head>
<body>
<div id="content"> </div> <button id="btn-fetch">Fetch Data</button>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script

type="text/javascript"> var btnFetch = document.getElementById('btn-
fetch'); var content = document.getElementById('content');

btnFetch.addEventListener('click', () => {
const xhr = new XMLHttpRequest();
xhr.open("GET", '/pet.json', true); xhr.onload = () => {
console.log(xhr.responseText); renderHtml(JSON.parse(xhr.responseText)); }
```

```
xhr.send();

}); function renderHtml(data) { content.innerHTML = ""; for (var i =
0; i <= data.length; i++) { let p = document.createElement('p'); let htmlpart =
""; htmlpart += data[i].name + " is a " + data[i].type + " with age" +
data[i].age + " years and weight " + data[i].weight + "kg and likes" +
data[i].likes; p.innerHTML = htmlpart; content.append(p); htmlpart = ""; } }
</script>
</body>
</html>
```

Practical7:- Store JSON data of few pets that you created in previous practical in a JSON file (copy
from console output of previous program to a .json file). Using AJAX, load data from the file and display
it in a presentable way using HTML and CSS.

Sol:-

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Practical 7</title>
<style type="text/css">
#pet-data {
border: 1px solid black;
border-radius: 10px;
border-collapse: collapse;
}
td {
border: 1px solid black;
border-collapse: collapse;
}
#btn-fetch {
margin-top: 20px;
font-size: 24px;
font-weight: bold;
```

```
background-color: black;
color: white;
border-radius: 8px;
}
</style>
</head>
<body>
<div id="content"> </div> <button id="btn-fetch">Fetch Data</button>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script

type="text/javascript"> var btnFetch = document.getElementById('btn-
fetch'); var content = document.getElementById('content');

btnFetch.addEventListener('click', () => {
const xhr = new XMLHttpRequest();
xhr.open("GET", '/pet.json', true); xhr.onload = () => {
console.log(xhr.responseText);
renderHtml(JSON.parse(xhr.responseText)); }
xhr.send();

}); function renderHtml(data) { content.innerHTML = ""; for (var i =
0; i <= data.length; i++) { let p = document.createElement('p'); let
htmlpart =
""; htmlpart += data[i].name + " is a " + data[i].type + " with age" +
data[i].age + " years and weight " + data[i].weight + "kg and likes" +
data[i].likes; p.innerHTML = htmlpart; content.append(p); htmlpart = "";
} }
</script>
</body>
</html>
```

Practical:-8 Create a plain HTML page for B.Sc. Hons CS course, mentioning details like fee, eligibility criteria, papers with names and credits, and future possibilities after the course. A button for styling should be there at bottom of the page. On clicking on this button JavaScript should redesign the complete page using jQuery in a nice presentable way.

Sol:-
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Practical 8</title>
<style type="text/css">
.container {
width: 70%;
margin: auto;
align-items: center;
background-color: #D9CAB3;
padding-bottom: 10px;
}
.info-table {
width: 80%;
margin: auto;
border: 3px solid black;
border-collapse: collapse;
margin-top: 2%;
margin-bottom: 2%;
}
.table-row {
width: 100%;
margin: auto;
}
.table-data {
width: 50%;
border: 2px solid white;
border-collapse: collapse;
}

</style>
</head>
<body>
<div>
<h1 class="heading">Bsc Hons Computer Science</h1>
<table>
<tr>
<td>Fee</td>
```

```
<td>25644</td>
</tr>
<tr>
<td>Eligibility Criteria</td>
<td>10-12 Pass</td>
</tr>
<tr>
<td>Subjects and credit scores</td>
<td>
<table>
<tr>
<th>Subject</th>
<th>Credit score</th>
</tr>
<tr>
<td>IT</td>
<td>6</td>
</tr>
<tr>
<td>Toc</td>
<td>6</td>
</tr>
<tr>
<td>DAV</td>
<td>4</td>
</tr>
<tr>
<td>DIP/Micro</td>
<td>4</td>
</tr>
</table>
</td>
</tr>
<tr>
<td>Future Opportunities</td>
<td>Paisa hi Paisa hoga</td>

</tr>
</table>
</div> <button id="btn-style"> Style Page </button>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script

type="text/javascript"> $(document).ready(function () { $('#btn-
style').click(function () { $("div").addClass('container');
```

```
$("table").addClass('info-table'); $("tr").addClass('table-row');
$("td").addClass('table-data'); $(".heading").css({ "textAlign": 'center' }); });
}); </script>
</body>
</html>
```

Practical:-9 Create an HTML page for an image gallery which shows the use of BOOTSTRAP to rearrange and resize its contents on resizing the browser.

Sol:-
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<title>Practical 9</title>
<style type="text/css">
/*img{ margin: 20px; }*/
</style>
</head>
<body>
<div class="jumbotron text-center">
<h1>IMAGE GALLERY</h1>
<p>Responsive Image gallery using bootstrap.</p>
</div>
<div class="container"> <img class="col-sm-4" src="https://picsum.photos/200/">
<img class="col-sm-4" src="https://picsum.photos/200/"><img/>
<img class="col-sm-4" src="https://picsum.photos/200/"><img/>
<img class="col-sm-4" src="https://picsum.photos/200/"><img/>
<img class="col-sm-4" src="https://picsum.photos/200/"><img/>
<img class="col-sm-4" src="https://picsum.photos/200/">
<img/>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
```

Practical10:- Create an HTTP server using Node.js which handles requests on port 10000 or a free port beyond 10000. Modify the server in such a way that opening localhost:10000 will display “Hello world, This is my Node.js server” on browser.

Sol:-
```
var http = require('http');
//create a server object:
http.createServer(function (req, res) {
res.write('hello ,this is my NOde.js server!'); //write a response to the client
```

```
res.end(); //end the response
}).listen(10000);
```

Practical11:-

Create index.html file containing two forms for SignIn and SignUp. Submitting SignIn form should search for credentials in mysql database using server created in previous practical. On successful signin, a welcome page should be displayed. Submitting SignUp form should insert new entry for credentials in mysql database using server created in previous practical. On successful signup, user should be returned
back to index.html.

SignUP.Html

Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
<script type="text/javascript" src="validate.js"></script>
<style>
* {
margin: 0;
padding: 0;
}
.ferror {
color: red;
}
.reset {
background: gray;
font-size: 24px;
color: black;
border-radius: 20px;
}
h1 {
text-align: center;
}
.sub {
background: rgb(0, 128, 75);
font-size: 24px;
color: black;
border-radius: 20px;
}
td {
```

```
font-size: 25px;
}
body{
background: rgb(58, 1, 1);
}
.form {
color: rgb(229, 228, 245);
align-items: center;
align-content: center;
font-size: 25px;
/* margin-left: 30px; */
}
.name2 {
width: 30%;
padding: 10px;
}
.center {
text-align: center;
justify-content: center;
align-items: center;
}
#gender {
margin-right: 200px;
}
.btn {
display: flex;
text-align: center;
align-items: center;
justify-content: center;
}
.sub {
margin-left: 100px;
}
.con{
margin-right: 456px;
}
@media only screen and (max-width:1080px){
.form{
display: flex;
flex-direction: column;
width: 100vw;
font-size: 20px;
}
.name2{
width: 70%;
}
```

```
#gender {
margin-right: 0px;
}
.con{
margin: 0;

}
.sub {
margin-left: 50px;
}
}
</style>
</head>
<body>
<div class="center">
<form action="/" name="myform" onsubmit="return validateform();" class="form"
method="post">

<table cellpadding="2" width="80%" align="center" cellspacing="2">
<h1>Sign UP Form</h1>
<div class="form" id="name">
User Name:
<input type="text" placeholder="Enter Your Name" name="name" id="name"

class="name2" size="30" required><b><span class="ferror"
class="name1"></span></b>

</div>
<br>
<div class="form" id="phone">
Phone:        
<input type="phone" placeholder="Enter Your Phone Number" name="phone"

id="phone" class="name2" size="30" required class="name1"><b><span

class="ferror"></span></b>

</div>
<br>
<div class="form" id="email">
Email Id:    
<input type="email" placeholder="Enter Your Email ID " name="email"

id="email" size="30" class="name2" required class="name1"><b><span

class="ferror"></span></b>
```

```
</div>
<br>
<div class="form" id="gender" class="gender">
Gender:    

required>

<input type="radio" name="gender" id="gender1" size="190" value="male"
Male
<input type="radio" name="gender" id="gender1" size="190" value="female">
Female
<b><span class="ferror"></span></b>
</div>
<br>
<div class="form" id="password">
Password:    

<input type="text" placeholder="Enter Your Password" name="password"

id="password" size="30" class="name2" required

class="name1"><b><span class="ferror"></span></b>
</div>
<br>
<div class="form" id="conpassword">
<div class="con">Confirmn</div>
Password:    
<input type="password" placeholder="ReEnter Your Password"
name="conpassword" id="conpassword1" size="30" class="name2" required
class="name1"><b><span class="ferror"></span></b>
</div>
<div class="form">
<input type="checkbox" name="Show Password" size="30"
id="Show Password" onclick="visibility()">
<label for="Show Password">Show Password</label><br></div>
<br><div class="btn" class="form">

class="reset"></div>

<div class="btn2"><input type="reset" name="reset"
<div class="btn1"></div><input type="submit" value="Sign UP"

class="sub" name="Sign Up">
</div>
</table>
```

```
</form>
</div>
</body>
<script>
function clearError() {
errors = document.getElementsByClassName("ferror");
for (let item of errors) {
item.innerHTML = "";
}
}
function seterror(id, error) {
element = document.getElementById(id);
element.getElementsByClassName("ferror")[0].innerHTML = error;
}
function validateform() {
clearError();
var returnval = true;
var name = document.forms["myform"]["name"].value;
if (name.length < 5) {
seterror("name", "*name is too short");
alert("*name is too short");
return false;
}
var phone = document.forms["myform"]["phone"].value;
if (phone.length != 10) {
seterror("phone", "*mobile number must have 10 digit");

alert("please enter valid phone number");
return false;
}
var email = document.forms["myform"]["email"].value;
if (email.length > 30 || email.length < 15) {
seterror("email", "*email is not more than 15b character");
alert("please enter valid email id");
return false;
}
var p=document.forms['myform']['password'].value;
if(p.length>=8
&&((p.includes('#'))||(p.includes('@'))||(p.includes('$'))||(p.includes('&'))||(p.includes('%'
))||(p.includes('.')))){
}
else{
alert('password Must contain special characte or length must be greater the 7');
return false;
}
var pasmatch=match_pass();
```

```
if(pasmatch){
}
else{
alert("Please Enter correct conpassword");
returnval = false;
return returnval
}
var f = confirm("YOU WANT TO SUMBIT A FORM");
if (f == true) {
var gender=document.forms['myform']['gender'].value;
returnval = true;
alert("your form submitted successfully")
console.log(gender)
console.log(name)
console.log(email)
console.log(phone)
console.log(p)
return returnval
}
else {
alert("your form not submitted ")
returnval = false;
return returnval
}
return returnval;
}
function visibility(){
var e=document.getElementById('conpassword1');

if(e.type==='password'){
console.log(e);
e.type='text';
}
else{
e.type="password";
}
match_pass()
}
function match_pass(){
var pass = document.forms["myform"]["password"].value;
var con_pass = document.forms["myform"]["conpassword1"].value;
console.log(pass," ",con_pass)
if(pass===con_pass){
return true;
}
else{
```

```
return false;
}
}
</script>
</html>
```

Sign In.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
<script type="text/javascript" src="validate.js"></script>
<style>
* {
margin: 0;
padding: 0;
}
.ferror {
color: red;
}
.reset {
background: gray;
font-size: 24px;
color: black;
border-radius: 20px;
}
h1 {
text-align: center;
}
.sub {
background: rgb(0, 128, 75);
font-size: 24px;
color: black;
border-radius: 20px;
}
td {
font-size: 25px;
}
body{
background: rgb(58, 1, 1);
}
.form {
color: rgb(229, 228, 245);
```

```
align-items: center;
align-content: center;
font-size: 25px;
/* margin-left: 30px; */

}
.name2 {
width: 30%;
padding: 10px;
}
.center {
text-align: center;
justify-content: center;
align-items: center;
}
#gender {
margin-right: 200px;
}
.form{
margin-top: 40px;
}
.btn {
display: flex;
text-align: center;
align-items: center;
justify-content: center;
}
.sub {
margin-left: 100px;
}
.con{
margin-right: 456px;
}
@media only screen and (max-width:1080px){
.form{
display: flex;
flex-direction: column;
width: 100vw;
font-size: 20px;
}
.name2{
width: 70%;
}
#gender {
margin-right: 0px;
}
```

```
.con{
margin: 0;
}
.sub {
margin-left: 50px;
}
}
</style>

</head>
<body>
<div class="center">
<form action="/sign" name="myform" onsubmit="return validateform();" class="form"
method="post">

<table cellpadding="2" width="80%" align="center" cellspacing="2">
<h1>Sign In form</h1>
<div class="form" id="email">
Email Id:    
<input type="email" name="email" placeholder="Enter Your Email Id"

id="email" size="30" class="name2" required class="name1"><b><span

class="ferror"></span></b>

</div>
<div class="form" id="conpassword">

Password:    
<input type="password" placeholder="Enter Your Password" name="password"

id="password"size="30" class="name2" required

class="name1"><b><span class="ferror"></span></b>
</div>
<div class="form1">
<input type="checkbox" name="Show Password" size="30"
id="Show Password" onclick="visibility()">
<label for="Show Password">Show Password</label><br></div>
<br><div class="btn" class="form">

<div class="btn1"></div><input type="submit" value="Sign In"

class="sub" name="Sign Up">
</div>
</table>
```

```
</form>
</div>
</body>
<script>
function visibility(){
var e=document.getElementById('password');
if(e.type==='password'){
console.log(e);
e.type='text';
}
else{
e.type="password";
}

var express = require('express');
var mysql = require('mysql')
var data;
var con = mysql.createConnection({
host: 'localhost',
user: 'root',
password: 'Himanshu123',
database: 'nodejs'
});
var app = express();
var bodyp = require('body-parser');
const send = require('send');
app.use(bodyp.json());
app.use(bodyp.urlencoded({
extended: true
}));
app.get('/', function (req, res) {
res.sendFile( dirname + '/itq11.html');
});
app.post('/', function (req, res) {
var name1 = req.body.name;
var phone = req.body.phone;
var email = req.body.email;
var gender = req.body.gender;
var p = req.body.password;
con.connect(function (err) {
if (err) throw err;
console.log("connect")
var sql = `Insert into formdata (name , phone , email_id , gender ,password ) values
('${name1}','${phone}','${email}','${gender}','${p}');`;
con.query(sql, function (err, result) {
if (err) {
```

```
res.redirect("/")
}
else {
res.redirect('/login')
}
Nodejs file
}
</script>
</html>

});
});
console.log(res.body)
});
app.get("/login", function (req, res) {
res.sendFile( dirname + '/signin.html');
});

app.post('/sign', function (req, res) {
var name1 = req.body.name;
var phone = req.body.phone;
var email = req.body.email;
email1 = req.body.email;
var gender = req.body.gender;
var p = req.body.password;
console.log("connect")
var sql = `select * from formdata where email_id='${email}' and password='${p}'`;
con.query(sql, function (err, result) {
data = JSON.parse(JSON.stringify(result));

email1 = data[0].name;
if (err) {
console.log(err)
res.redirect("/login")
}
else {
res.redirect('/project')

});
}
});

app.get("/project", function (req, res) {
console.log(data[0].name);
console.log(data[0].phone);
console.log(data[0].gender);
```

```
console.log(data[0].email_id);
// res.sendFile( dirname +'/responsive.html');
res.end(`<h1>             
            
            
               &
nbspwelcome
User</h1><br>`+`<h2>Name:  
            
             
${data[0].name}</h2><br>`
+`<h2>Gender:  
               &
nbsp  &nbs
p    ${data[0].gender}</h2><br>`
```