

Course Code/ Course Subject: U18CSI6201L INTERNET & WEB PROGRAMMING LAB

S.No	Table of Contents	Remarks
1.	Develop a webpage using HTML. (Student profile)	
2.	Apply style specification in HTML page using CSS.	
3.	Develop a HTML form and validate it using Java script.	
4.	Random over and Rollover using DOM javascript	
5.	Demonstrate exception handling using Java Script.	
6.	Develop an XML document and validate it using SCHEMA.	
7.	Develop an XML document and transform it into HTML using XSLT.	
8.	Develop a servlet program to add two numbers.	
9.	Develop a JSP form to collect user registration details.	
10.	Develop a JSP login form with cookies.	
11.	Apply JavaBean class to print information about a student class.	
12.	For Front end webpage connect with backend database and perform simple CRUD operations using JSF, PHP etc.	

1. Develop a webpage using HTML. (Student profile).
2. Apply style specification in HTML page using CSS.

HTML Code:

a) Index.html:

```
<!DOCTYPE html>
<html>
<head>
<title>Document</title>
<style>
  body {
    margin: 0;
    font-family: Arial, Helvetica, sans-serif;
    color: #fff;
    text-align: left;
  }

  .topnav {
    overflow: hidden;
    background-color: #333;
  }

  .topnav a {
    float: left;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
    font-size: 17px;
  }

  .topnav a:hover {
    background-color: red;
    color: black;
  }
  h2{
    text-decoration: underline;
  }
```

```

img{
  width:300px;
  height:300px;
  border-radius: 200px;
}
</style>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="styles.css" />
</head>

<body>
<div class="topnav">
  <a href="#home">Home</a>
  <a href="./skills.html">Skills</a>
  <a href="./contact.html">Contact</a>
  <a href="./about.html">About</a>
</div>
<table>
<tr>
<td>
<table>
<tr>
<th>

<h2>About Me</h2>
<p style="padding-right: 35rem;">
  My Name is Surya Prakash and my experience in working with web
development tools like Django, Mongo DB, MySQL has developed a great
sense of confidence at a personal level. Hence, I seek a challenging
position in areas of full stack development where I can
provide for the organization and expand my capabilities further in
the pursuit of career advancement.
</p>
<br/>
<h2>Skills</h2>
<h4>Languages : Python,C,HTML,CSS,Javascript,SQL</h4>
<h4>
Tools : Code Blocks, Visual Studio, Xampp Server

```

</h4>

<h4>

Technologies : Machine Learning, Deep Learning, Web
Development

</h4>

</th>

</tr>

<tr>

<th>

<hr />

<h2>Soft Skills</h2>

<h4>Problem Solver</h4>

<h4>Good Adaptability</h4>

<h4>Good Communication & Team Work</h4>

<h4>Languages Known: English, Tamil</h4>

<hr />

</th>

</tr>

<tr>

<th>

<h2>Certifications</h2>

<h4>Microsoft Azure : AI Fundamentals</h4>

<h4>Crash Course on Python</h4>

<h4>Google's The Fundamentals of Digital Marketing</h4>

<h4>Responsive Web Design(freecodecamp)</h4>

<hr />

</th>

</tr>

<tr>

<th>

<h2>Hobbies</h2>

<h4>Console Gaming</h4>

<h4>Speed Cubing</h4>

</th>

</tr>

<tr>

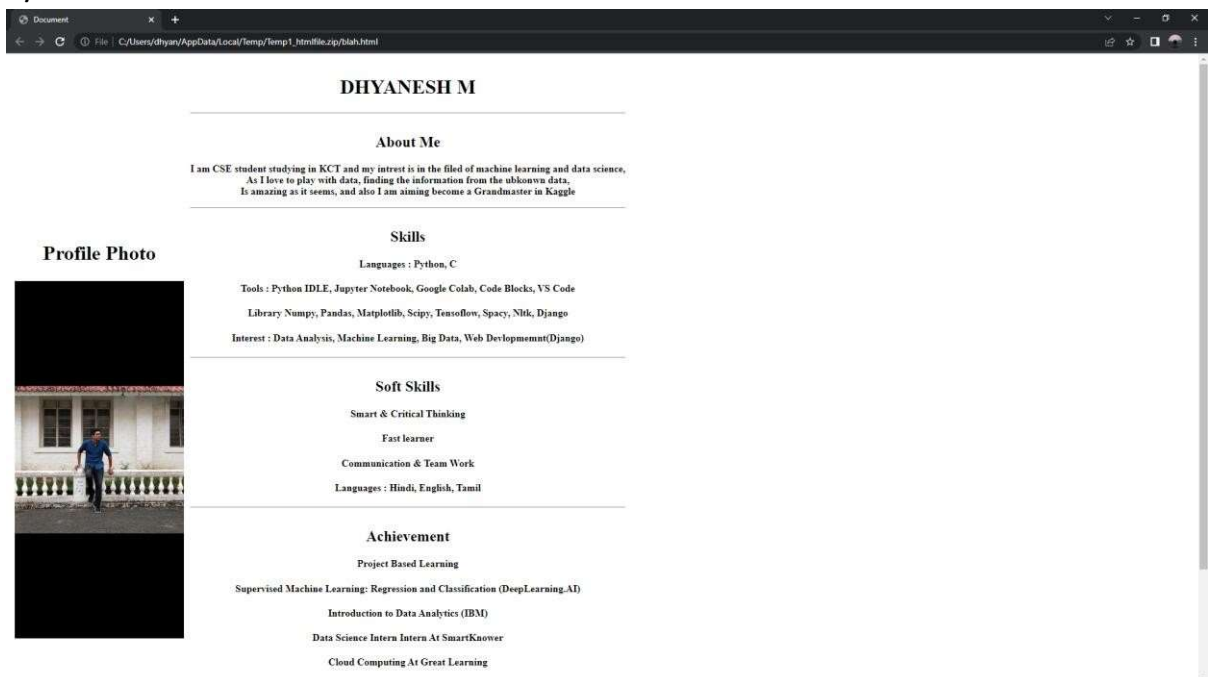
<th>

<hr />

```

<h1>Contact Me</h1>
<form>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" /><br /><br />
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" /><br /><br />
  <label for="message">Message:</label>
  <textarea id="message" name="message"></textarea><br /><br />
  <input type="submit" value="Submit" />
</form>
</th>
</tr>
<tr>
<th>
  <audio src="Bell.mp3" controls></audio>
</th>
</tr>
</table>
</td>
</tr>
</table>
</body>
</html>

```



b) Skills.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>
  <style>
    body {
      margin: 0;
      font-family: Arial, Helvetica, sans-serif;

      color: #f2f2f2;
    }

    .topnav {
      overflow: hidden;
      background-color: #333;
    }

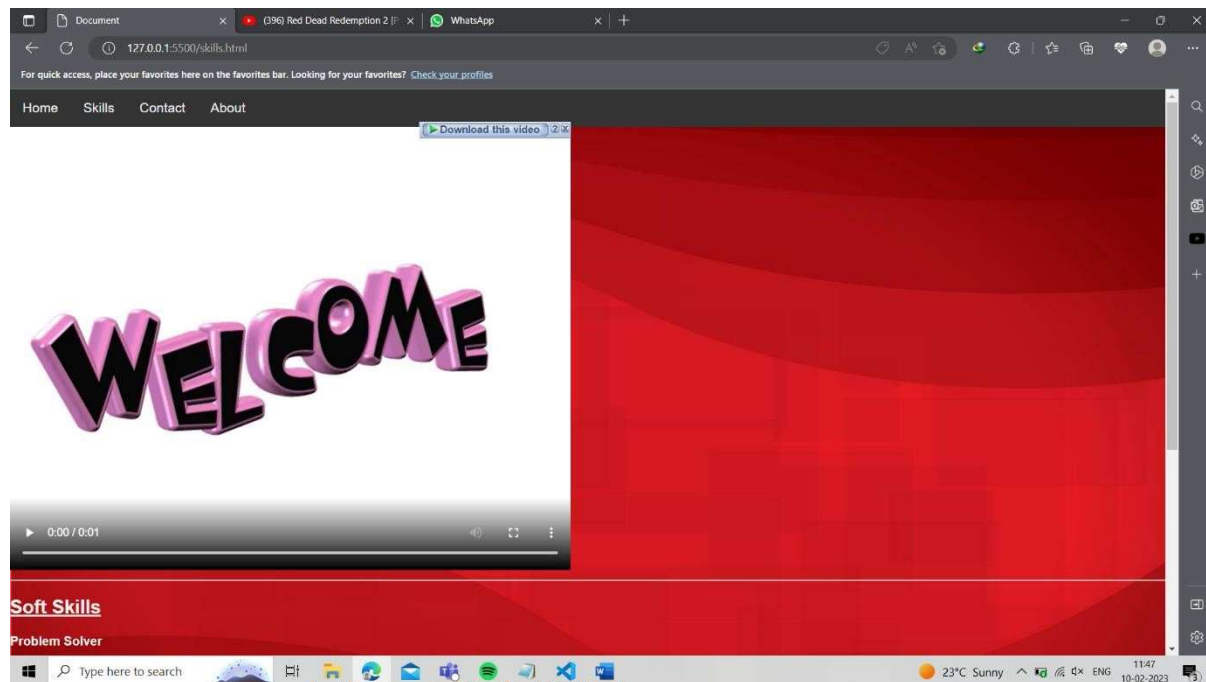
    .topnav a {
      float: left;
      color: #f2f2f2;
      text-align: center;
      padding: 14px 16px;
      text-decoration: none;
      font-size: 17px;
    }

    .topnav a:hover {
```

```
background-color: red;
color: black;
}

h2{
text-decoration: underline;
}
</style>
<link rel="stylesheet" href="styles.css" />
</head>
<body>
<div class="topnav">
  <a href="./index.html">Home</a>
  <a href="#">Skills</a>
  <a href="./contact.html">Contact</a>
  <a href="./about.html">About</a>
</div>
<video src="./images/new1.mp4" controls width:100%></video>
<hr/>
<tr>
  <th>
    <h2>Soft Skills</h2>
    <h4>Problem Solver</h4>
    <h4>Good Adaptability</h4>
    <h4>Good Communication & Team Work</h4>
    <h4>Languages Known: English, Tamil</h4>
    <hr />
  </th>
</tr>
<tr>
  <th>
    <h2>Certifications</h2>
    <h4>Microsoft Azure : AI Fundamentals</h4>
    <h4>Crash Course on Python</h4>
    <h4>Google's The Fundamentals of Digital Marketing</h4>
    <h4>IBM Machine Learning</h4>
    <h4>Intro to Tensorflow for Deep Learning</h4>
    <hr />
  </th>
</tr>
```

```
</th>
</tr>
</body>
</html>
```



c) Contact.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>

  <style>

    body {

      margin: 0;

      font-family: Arial, Helvetica, sans-serif;
```



```
    color:white;
}
```

```
.topnav {
    overflow: hidden;
    background-color: #333;
}
```

```
.topnav a {
    float: left;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
    font-size: 17px;
}
```

```
.topnav a:hover {
    background-color: red;
    color: black;
}
```

```
h1{
    text-decoration: underline;
}
```

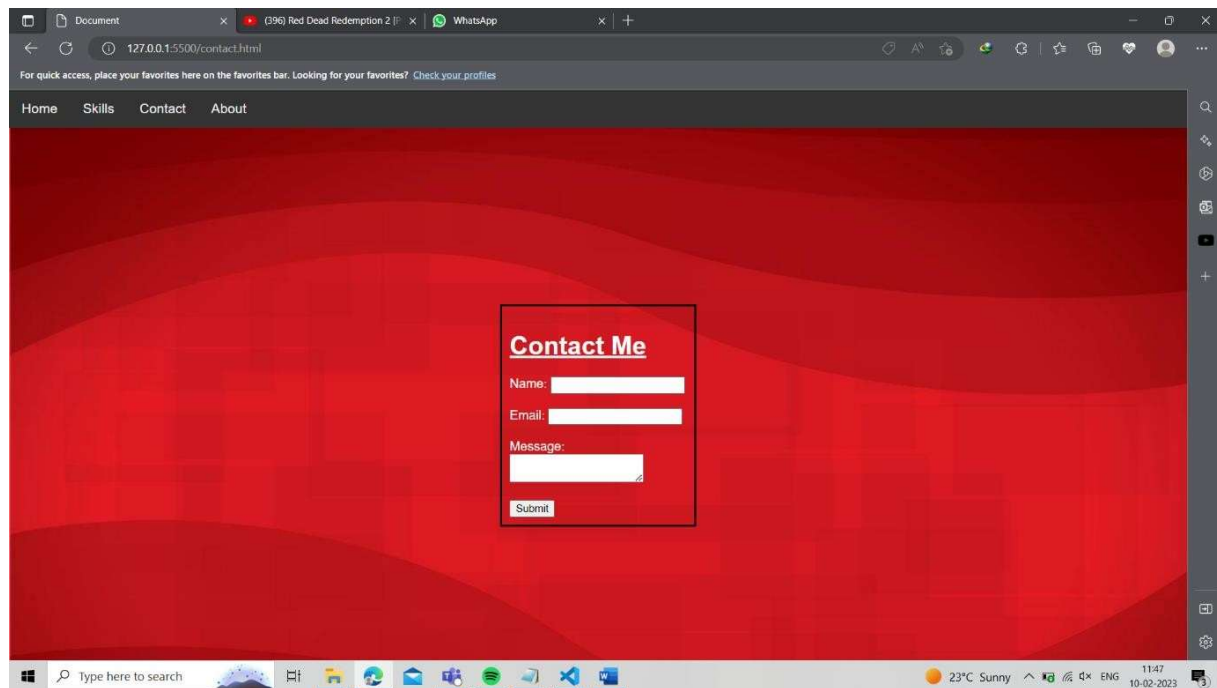
```
.new{
    margin:auto;
    width: 15%;
```

```
border:3px solid black;
padding:10px;
margin-top:15%;
}
</style>
<link rel="stylesheet" href="styles.css" />
</head>
<body>
<div class="topnav">
  <a href="./index.html">Home</a>
  <a href="./skills.html">Skills</a>
  <a href="#">Contact</a>
  <a href="./about.html">About</a>

</div>
<div class="new">
<h1>Contact Me</h1>
<form>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" /><br /><br />
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" /><br /><br />
  <label for="message">Message:</label>
  <textarea id="message" name="message"></textarea><br /><br />
  <input type="submit" value="Submit" />
</form>
</div>
```

</body>

</html>



d) About.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>
```

```
<style>
```

```
body {
```

```
margin: 0;
```

```
font-family: Arial, Helvetica, sans-serif;
```

```
color:#f2f2f2;
```

```
}
```

```
.topnav {
```

```
overflow: hidden;
```

```
background-color: #333;
```

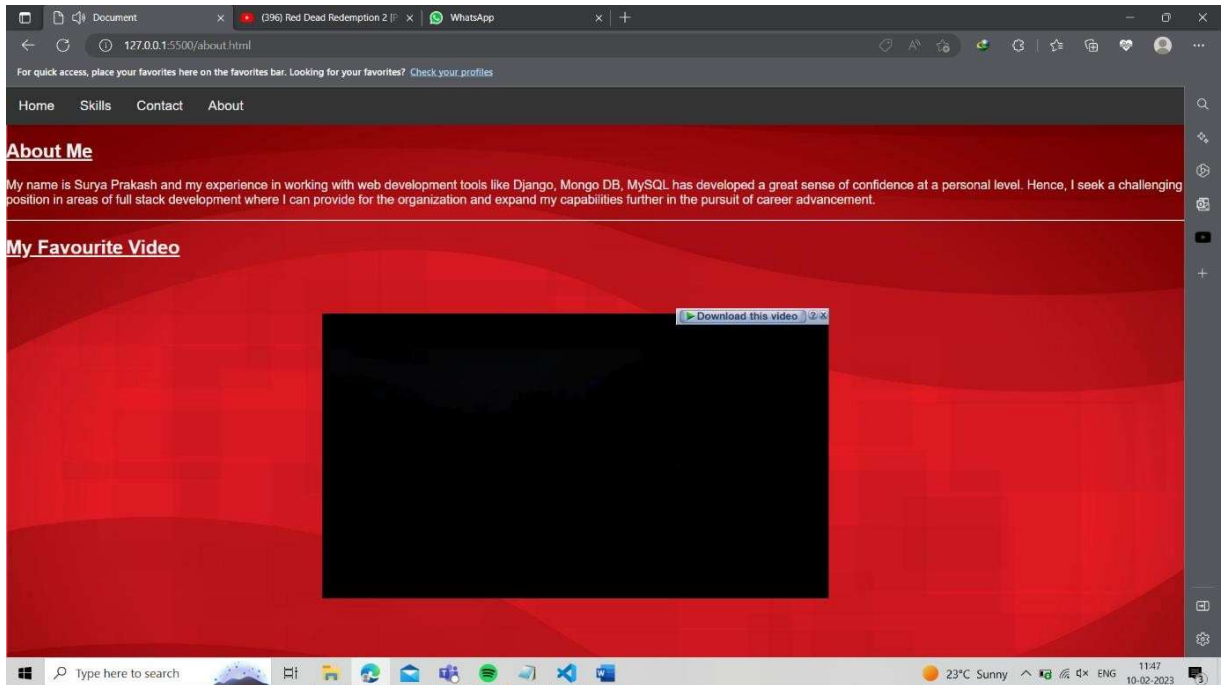
```
}
```

```
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: red;
  color: black;
}
h2{
  text-decoration: underline;
}
#vid{
  margin-left: 400px;
  margin-top: 50px;
}
</style>
<link rel="stylesheet" href="styles.css" />
</head>
<body>
  <div class="topnav">
    <a href="/index.html">Home</a>
    <a href="/skills.html">Skills</a>
    <a href="/contact.html">Contact</a>
    <a href="#">About</a>
  </div>
  <h2>About Me</h2>
  <p>
    My name is Surya Prakash and my experience in working with web
    development tools like Django, Mongo DB, MySQL has developed a great
    sense of confidence at a personal level. Hence, I seek a challenging
    position in areas of full stack development where I can
    provide for the organization and expand my capabilities further in the
    pursuit of career advancement.
  </p>
```

```

</hr>
<h2>My Favourite Video</h2>
<video src="./images/rdr2.mp4" controls id="vid"></video>
</body>
</html>

```



e) Project.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>
  <link rel="stylesheet" href="styles.css">
  <style>
    body {
      margin: 0;
      font-family: Arial, Helvetica, sans-serif;
      color:#f2f2f2;
    }

    .topnav {

```

```
overflow: hidden;
background-color: #333;
}
```

```
.topnav a {
float: left;
color: #f2f2f2;
text-align: center;
padding: 14px 16px;
text-decoration: none;
font-size: 17px;
}
```

```
.topnav a:hover {
background-color: red;
color: black;
}
```

```
h2{
text-decoration: underline;
}
```

```
.fixed{
position: fixed;
bottom: 0;
right: 0;
width: 300px;
border: 3px solid #000;
}
```

```
.absolute{
position: absolute;
top: 450px;
right: 500px;
bottom: 0;
width: 500px;
border: 3px solid #000;
}
```

```

    .img{
      width:300px;
      height:300px;
      border-radius: 200px;
    }
    .img:hover{
      -webkit-transform: rotate(360deg);
      -webkit-transition-duration: 1s;
      -webkit-transition-delay: now;
      -webkit-animation-timing-function: linear;
      -webkit-animation-iteration-count: infinite;
    }
    .flex{
      display: flex;
      flex-wrap: nowrap;
      justify-content: space-between;
    }
    .flex-container > img {
      position: fixed;
      width: 100px;
      margin: 10px;
      right: 0;
    }
  </style>
</head>
<body>
  <div class="topnav">
    <a href="/index.html">Home</a>
    <a href="/skills.html">Skills</a>
    <a href="/project.html">Projects</a>
    <a href="/contact.html">Contact</a>
    <a href="#">About</a>
  </div>
  <h1 style="text-decoration: underline;">My Projects</h1>
  <div class="flex">
    
    
    

```

```
</div>
<div class="fixed">
  <h2>Project 1</h2>
  <h2>Fitness Management Site</h2>
  <h3>It Is Course based project where the fitness videos were merged and
provided in the course and BMI report and diet chart is
  generated according to the BMI Value</h3>
</div>
<div class="absolute">
  <h2>Project 2</h2>
  <h2>Estimating Bird Population using Deep Audio Classification</h2>
  <h3>It Is project which estimate the bird's population where the audio file
is converted into waveform, waveform into spectrogram,
  three minute audio file is split into 3 seconds and the bird population will
be predicted according to the bird call.</h3>
</div>
</body>
</html>
```

Styles.css:

```
body{
  background: url(/images/new.jpg);
  background-repeat: no-repeat;
  background-size: cover;
  height: 100vh;
  background-size: cover;
  background-attachment: fixed;
}
```


My Projects



Project 2

Estimating Bird Population using Deep Audio Classification

It is project which estimate the bird's population where the audio file is converted into waveform, waveform into spectrogram, three minute audio file is split into 3 seconds and the bird population will be predicted according to the bird call.

Project 1

Fitness Management Site

It is Course based project where the fitness videos were merged and provided in the course and BMI report and diet chart is generated according to the BMI Value

3. Develop a HTML form and validate it using Java script.

Validation Form JS.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>Form</title>

    <link rel="stylesheet" href="validation form js styles.css">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link
href="https://fonts.googleapis.com/css2?family=Oleo+Script+Swash+Caps&family=Yanone+Kaffeesatz&display=swap" rel="stylesheet">

    <script>

        function prom() {

            const crno=prompt("Enter your College Registration Number:");

        }

        function validate()

        {

            const name=document.querySelector('#name').value;

            const email=document.querySelector('#email').value;

            var atposition=email.indexOf("@");

            var dotposition=email.lastIndexOf(".");

            const phno=document.querySelector('#phno').value;

            const p=/^\d{10}$/;
```

```
const dept=document.querySelector('#dept').value;
const coll=document.querySelector('#coll').value;
const pst=document.querySelector('#pst').value;
if(name==" " || name==null)
{
    alert("Name not Entered!");
    return false;
}
if (atposition<1 || dotposition<atposition+2 ||
dotposition+2>=email.length)
{
    alert("Please Enter a Valid E-mail Address");
    return false;
}
if (phno.length==10)
{
    if (phno.match(p))
    {
        console.log("");
    }
    else
    {
        alert("Enter a Valid Phone Number!");
        return false;
    }
}
else
```

```
{  
    alert("Enter a Valid Phone Number!");  
    return false;  
}  
  
if (dept=="" || dept==null) {  
    alert("Department not Selected!");  
    return false;  
}  
  
if (coll=="" || coll==null) {  
    alert("College Name not Entered!");  
    return false;  
}  
  
if (pst=="" || pst==null){  
    alert("Problem Statement not Entered!");  
    return false;  
}  
  
else  
{  
    if (pst.length>60) {  
        console.log("");  
    }  
    else  
    {  
        alert("Problem Statement too short!");  
        return false;  
    }  
}
```

```
        }
    }
    if (confirm("Are your entered details correct?: ") == true) {
        return true;
    }
    else {
        return false;
    }
    if(tno==null){
        alert("Please specify the No.of Team Members");
        return false;
    }
}
</script>
</head>
<body onload="prom()">
    <h1>Hackathon Registration</h1>
    <div id="form_div">

        <form action="#" onsubmit="return validate();">
            <p>
                <label for="name">Name</label>
                <input type="text" id="name" placeholder="Enter Full Name">
            </p>
            <p>
                <label for="email">Email</label>
```

```
<input type="text" id="email" placeholder="Enter College mail-id">
</p>
<p>
  <label for="phno">Phone No</label>
  <input type="text" id="phno" placeholder="Enter Phone No">
</p>
<p>
  <label for="dept">Department</label>
  <select id="dept">
    <option value="">Select Department</option>
    <option value="CSE">CSE</option>
    <option value="ISE">ISE</option>
    <option value="AIDS">AIDS</option>
    <option value="IT">IT</option>
  </select>
</p>
<p>
  <label for="coll">College</label>
  <input type="text" id="coll" placeholder="Enter College Name">

</p>
  <label for="coll">Domain &nbsp;</label>
  <input type="radio" id="frontend" name="fav_language"
value="frontend">
  <label for="html">Frontend(HTML,CSS,ReactJS)</label>
  <input type="radio" id="backend" name="fav_language"
value="backend">
```

```

        <label for="css">Backend(NodeJS,Django,Flask)</label>
    <p>
        <label for="coll">No.of Team Members:</label>
        <input type="number" id="tno" min=0 max=10>
    </p>
    <label for="coll">Accompanied By? &nbsp;  </label>
    <input type="radio" id="frontend" name="fav_language"
value="frontend">
    <label for="html">College</label>
    <input type="radio" id="backend" name="fav_language"
value="backend">
    <label for="css">Self</label>
    <p>
        <label for="pst">Problem Statement</label>
        <br>
        <textarea id="pst" placeholder="Enter choosen problem
statement"></textarea>
    </p>
    <p>
        <button id="bon" style="margin-left: 100px;">Submit</button>
    </p>
</form>
</div>
</body>
</html>

```

Validation Form JS Styles.css:

```
body{
```

```
font-family: sans-serif;
font-size: 10px;
background:lightblue;
background-size: cover;
background-position:0;
}
```

```
h1{
  text-align: center;
}
```

```
#form_div{
  background:url(images/fobackn.jpg);
  width: 400px;
  height: 430px;
  border-style: inset;
  border-color: black;
  border-radius: 10px;
  border-width: 4px;
  display: flex;
  align-items: center;
  justify-content: center;
  margin-left: 420px;
}
```

```
button{
  text-align: center;
```



```
font-family: 'Yanone Kaffeesatz', sans-serif;  
font-weight: 600;  
width: 90px;  
height: 40px;  
font-size: 20px;  
border-color: black;  
transition: 1s;  
}
```

```
#name{  
    margin-left: 40px;  
    width: 220px;  
}
```

```
#email{  
    margin-left: 43px;  
    width: 220px;  
}
```

```
#phno{  
    margin-left: 13px;  
  
    width: 220px;  
}
```

```
#dept{  
    margin-left: 4px;
```

```
    width: 220px;  
}
```

```
#coll{  
    margin-left: 29px;  
    width: 220px;  
}
```

```
#pst{  
    width:305px;  
    height:70px;  
}
```

```
#bon{  
    margin-bottom: 10px;  
}
```

```
#bon:hover{  
    background-color: aquamarine;  
    color: black;  
}
```

Hackathon Registration

Name

Enter Full Name

Email

Enter College mail-id

Phone No

Enter Phone No

Department

Select Department

College

Enter College Name

Domain

☐ Frontend(HTML,CSS,ReactJS)

☐ Backend(NodeJS,Django,Flask)

No of Team Members

Accompanied By?

☐ College

☐ Self

Problem Statement

Enter choosen problem statement

Submit

4. Random over and Rollover using DOM Javascript.

5. Demonstrate exception handling using Java Script.

Index.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>Javascript Rollover</title>
  <style>
    body{
      background-image: url('new.jpg');
    }

    img{
      height: 200px;
      width: 300px;
      border-radius: 10px;
    }

    #tag{
      visibility: hidden;
      font-size: larger;
      font-weight: 700;
    }

    input{
      padding: 20px;
```

```
border-radius: 10px;  
border: none;  
background-color: rgb(255, 255, 255, 0.1);  
}
```

```
button{  
padding: 10px;  
border-radius: 10px;  
display: block;  
background-color: rgb(255, 255, 255, 0.8);  
border: none;  
margin-top: 10px;  
}
```

```
form{  
background-color: rgb(255, 255, 255, 0.1);  
padding: 30px;  
  
border-radius: 10px;  
width: 200px;  
border: 2px solid black;  
}
```

```
hr{  
border: 2px solid white;  
border-radius: 10px;  
width: 40%;
```

```
    opacity: 0.3;
    margin: 60px 0px;
}
```

```
.col{
    float: left;
    width: 50%;
}
```

```
.row:after {
    content: "";
    display: table;
    clear: both;
}
```

```
#message{
    height: fit-content;
    padding: 10px;

    border-radius: 10px;
    margin-top:0;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
    <div class="row">
```

```
        <div class="col">
```

```
<h3>
```

```
    Image Rollover Event
```

```
</h3>
```

```

```

```
<p id="tag">
```

```
    Image is getting hovered right now!
```

```
</p>
```

```
<hr>
```

```
</div>
```

```
<div class="col">
```

```
    <div class="samp">
```

```
        <form onsubmit="tryCatch()" id="myform">
```

```
            <h3>
```

```
                Try Catch
```

```
            </h3>
```

```
            <label for="age">
```

```
                Enter the Protagonist Name <br><br>
```

```
                <input type="text" id="name" placeholder="enter the name"
name="name">
```

```
            </label>
```

```
            <button type="submit" >submit</button>
```

```
        </form>
```

```
        <p id="message">
```

```
        </p>
```

```
    </div>
```

```
</div>
```

```
</div>
```

```
<script>
```

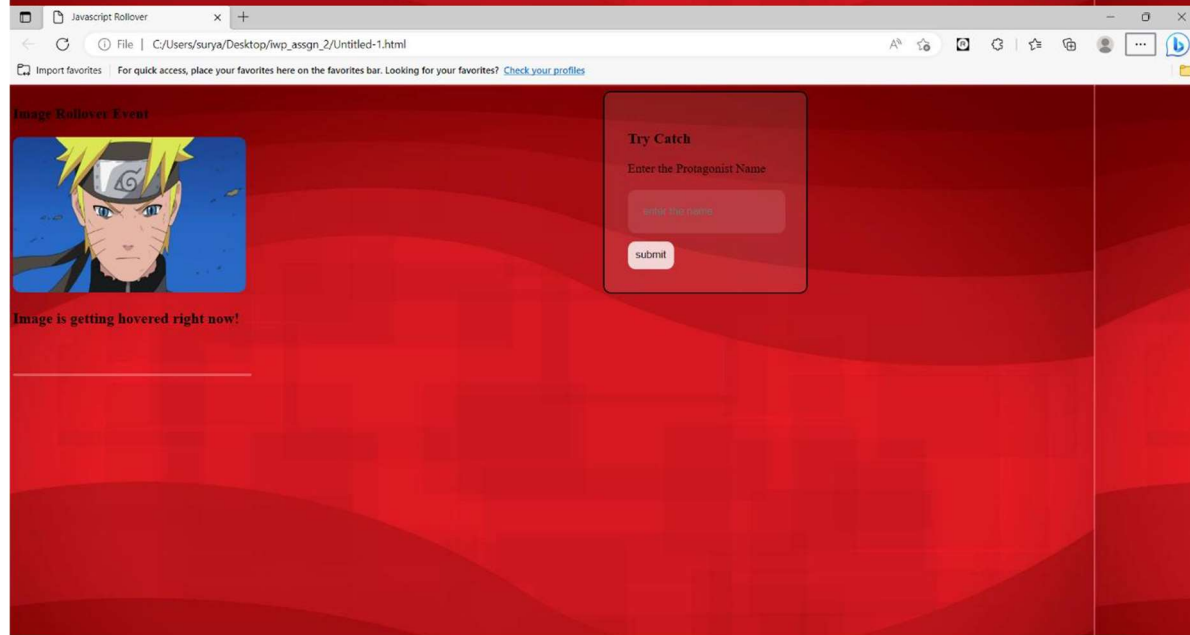
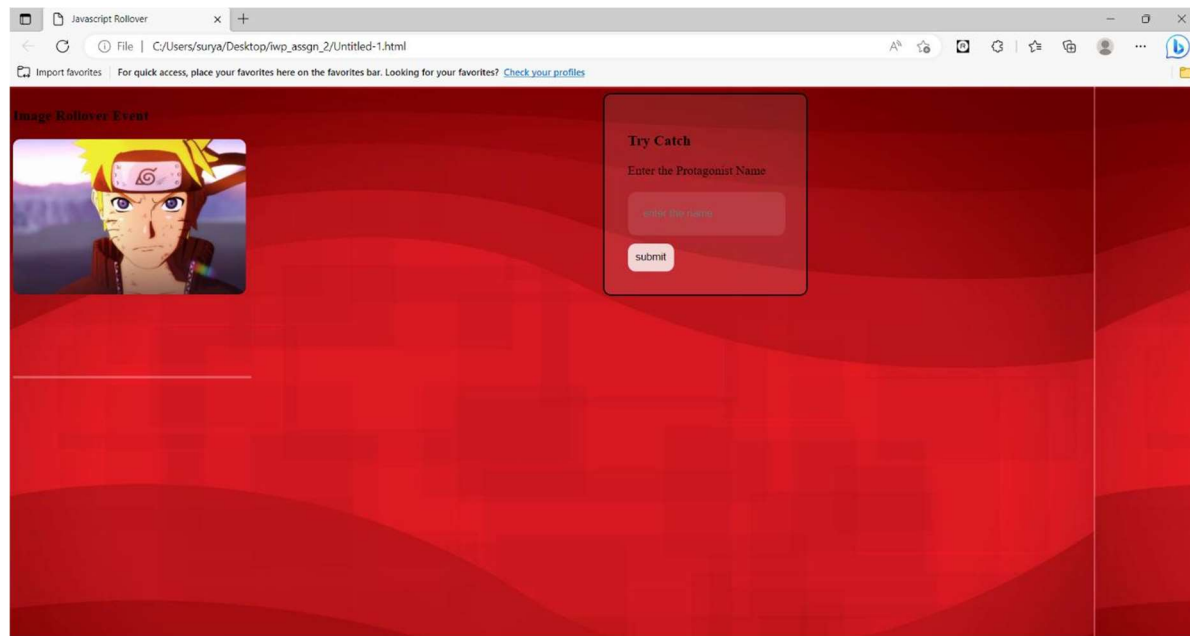
```
var Img = document.getElementById('img');
var tag = document.getElementById('tag');
var form = document.getElementById('myform');
Img.addEventListener('mouseover', function(){
    Img.src = 'img2.jpg';
    tag.style.visibility = 'visible';
})
Img.addEventListener('mouseleave', function(){
    Img.src = 'img1.jpg';
    tag.style.visibility = 'hidden';
})
```

```
function tryCatch(){
    var name = document.getElementById('name').value;
    var message = document.getElementById('message');
    try{
        if(name == "") throw "Empty";

        const specialChars = /[`!@#$%^&*()_+\-=\[\]{};':"\\|,.<>V/?~]/;
        if (specialChars.test(name)) throw "Not a valid string";
        const num = /\d+\/;
        if (num.test(name)) throw "Not a valid string";
        if(name=="Naruto" || name=="naruto") throw "is right";
        if(name!="Naruto" || name!="naruto") throw "is not right";
    }catch(errMsg){
```



```
        message.innerHTML = "Your entered input "+errMsg;  
    }  
  
    }  
    form.addEventListener('submit', function(event){  
        event.preventDefault();  
    })  
</script>  
</body>  
</html>
```



6. Develop an XML document and validate it using SCHEMA.

Student.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<students xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema.xsd">
  <student>
    <name>Surya</name>
    <dept>CSE</dept>
    <address>Coimbatore</address>
  </student>
  <student>
    <name>Dinax</name>
    <dept>CSE</dept>
    <address>Hosur</address>

  </student>
  <student>
    <name>Vasanth</name>
    <dept>CSE</dept>
    <address>Erode</address>
  </student>
  <student>
    <name>Allwin</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
  <student>
    <name>Mahin</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Vishwa</name>
    <dept>CSE</dept>
    <address>Dindugul</address>
  </student>
  <student>
    <name>Vishnu</name>
```

```

    <dept>CSE</dept>
    <address>Kerala</address>
</student>
<student>
    <name>Vignesh</name>
    <dept>CSE</dept>
    <address>Trichy</address>
</student>
<student>
    <name>Dhanush</name>
    <dept>CSE</dept>
    <address>Delhi</address>
</student>
<student>
    <name>Karthi</name>
    <dept>CSE</dept>
    <address>Erode</address>
</student>
</students>

```

Schema.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="students">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="student" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string" />
              <xs:element name="dept" type="xs:string" />
              <xs:element name="address" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

</xs:schema>

The image displays two windows of the XML Copy Editor application. The top window, titled 'student.xml - XML Copy Editor', shows an XML document with 44 lines. The root element is 'students', which contains a sequence of 'student' elements. Each 'student' element has three children: 'name', 'dept', and 'address'. The 'name' elements include: Dinesh, Vasanth, Abhishek, Mahesh, Vishwas, Dindugudi, Vishnu, Krishna, Vignesh, and Trishy. The 'dept' elements are all 'CSE'. The 'address' elements are: Bangalore, Chennai, Bangalore, Dindugudi, Bangalore, Chennai, Bangalore, Chennai, Bangalore, and Chennai. The bottom window, titled 'schema.xsd - XML Copy Editor', shows the XSD schema for the 'students' XML. The schema defines a root element 'students' of type 'xs:sequence' containing one or more 'student' elements. The 'student' element is defined as a complex type with three children: 'name', 'dept', and 'address', all of type 'xs:string'. The schema is valid, as indicated by the status bar.

```
<?xml version="1.0" encoding="UTF-8"?>
<students xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="schema.xsd">
  <student>
    <name>Dinesh</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Vasanth</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
  <student>
    <name>Abhishek</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Mahesh</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
  <student>
    <name>Vishwas</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Dindugudi</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
  <student>
    <name>Vishnu</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Krishna</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
  <student>
    <name>Vignesh</name>
    <dept>CSE</dept>
    <address>Bangalore</address>
  </student>
  <student>
    <name>Trishy</name>
    <dept>CSE</dept>
    <address>Chennai</address>
  </student>
</students>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="students">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="student" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string" />
              <xs:element name="dept" type="xs:string" />
              <xs:element name="address" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

7. Develop an XML document and transform it into HTML using XSLT.

Student-2.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="stdntstyle-1.xsl"?>
<students>
  <student>
    <name>
      Surya
    </name>
    <dept>
      CSE
    </dept>
    <mark>
      99
    </mark>
    <exam>
      ielts
    </exam>
    <placement>
      <company>
        flipr
      </company>
      <package>
        13lpa
      </package>
    </placement>
  </student>
  <student>
    <name>
      Vasanth
    </name>
    <dept>
      ISE
    </dept>
    <mark>
      90
    </mark>
    <exam>
```

```
        dom
    </exam>
    <placement>
        <company>
            forge
        </company>
        <package>
            12lpa
        </package>
    </placement>
</student>
<student>
    <name>
        Mahin
    </name>
    <dept>
        Mech
    </dept>
    <mark>
        92
    </mark>
    <exam>
        ielts
    </exam>
    <placement>
        <company>
            Amazon
        </company>
        <package>
            10lpa
        </package>
    </placement>
</student>
</students>
```

Stdntstyle-1.xsl:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xsl:stylesheet version="1.0"
```

```
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="/">
```

```
  <html>
```

```
  <head>
```

```
  </head>
```

```
  <body>
```

```
    <h2>Students Data- XML to HTML using XSLT</h2>
```

```
    <table border="1">
```

```
      <tr bgcolor="gray">
```

```
        <th>name</th>
```

```
        <th>dept</th>
```

```
        <th>mark</th>
```

```
        <th>exam</th>
```

```
      </tr>
```

```
      <xsl:for-each select="students/student">
```

```
        <tr>
```

```
          <td><xsl:value-of select="name"/></td>
```

```
          <td><xsl:value-of select="dept"/></td>
```

```
          <td><xsl:value-of select="mark"/></td>
```

```
          <td><xsl:value-of select="exam"/></td>
```

```
        </tr>
```

```
      </xsl:for-each>
```

```
    </table>
```

```
  </body>
```

```
</html>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```


student-2.xml - XML Copy Editor

File Edit View Insert XML Tools Help

studentstyle-1.xsl X student-2.xml X

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/xsl" href="studentstyle-1.xsl"?>
3 <students>
4 <student>
5   <name>
6     Sanya
7   </name>
8   <dept>
9     CSE
10  </dept>
11  <mark>
12    99
13  </mark>
14  <exam>
15    <placement>
16      <company>
17        TCS
18      </company>
19      <package>
20        12pa
21      </package>
22    </placement>
23  </exam>
24 </student>
25 <student>
26   <name>
27     Varadha
28   </name>
29   <dept>
30     ISE
31   </dept>
32   <mark>
33     90
34   </mark>
35   <exam>
36     <placement>
37       <company>
38        TCS
39      </company>
40      <package>
41        12pa
42      </package>
43    </placement>
44  </exam>
45 </student>
46 <student>
47   <name>
48     Mahin
49   </name>
50 </student>
51 </students>
```

Current Element: student

Insert Element: dept, exam, mark, name, placement

Insert Sibling: student

Insert Entity: amp, apos, gt, lt, quot

Ln 10 Col 16

studentstyle-1.xsl - XML Copy Editor

File Edit View Insert XML Tools Help

studentstyle-1.xsl X student-2.xml X

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet href="studentstyle-1.xsl" type="text/xsl"?>
3 <xsl:output method="html" encoding="UTF-8" indent="yes"/>
4 <xsl:template match="/">
5   <html>
6     <head>
7       <title>
8         Students Data
9       </title>
10    </head>
11    <body>
12      <table border="1">
13        <thead>
14          <tr>
15            <th>name</th>
16            <th>dept</th>
17            <th>mark</th>
18            <th>exam</th>
19          </tr>
20        </thead>
21        <xsl:for-each select="students/student">
22          <tr>
23            <td><xsl:value-of select="name"/></td>
24            <td><xsl:value-of select="dept"/></td>
25            <td><xsl:value-of select="mark"/></td>
26            <td><xsl:value-of select="exam"/></td>
27          </tr>
28        </xsl:for-each>
29      </table>
30    </body>
31  </html>
32 </xsl:template>
33 </xsl:stylesheet>
```

Current Element: html

Insert Element: body, head

Insert Sibling: html

Insert Entity: amp, apos, gt, lt, quot

Ln 9 Col 10

127.0.0.1:5500/student-2.xml

127.0.0.1:5500/student-2.xml

Students Data- XML to HTML using XSLT

name	dept	mark	exam
Surya	CSE	99	ielts
Vasanth	ISE	90	dom
Mahin	Mech	92	ielts

8. Develop a servlet program to add two numbers.

index.html

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>TODO supply a title</title>
```

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

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

```
  </head>
```

```
  <body>
```

```
    <h1>Enter two numbers for addition...</h1>
```

```
    <form action="add" method="POST">
```

```
      Number 1:<input type="text" id="num1"
```

```
      name="num1"><br><br>Number 2:<input
```

```
      type="text" id="num2" name="num2"><br><br>
```

```
      <input type="submit">
```

```
    </form>
```

```
  </body>
```

```
</html>
```

add.java

import

java.io.IOException;

import

java.io.PrintWriter;

import

javax.servlet.ServletException;

import

javax.servlet.annotation.WebServlet;

import

javax.servlet.http.HttpServlet;

import

javax.servlet.http.HttpServletRequest;

import

javax.servlet.http.HttpServletResponse;

@WebServlet("/add")

public class add extends HttpServlet {

public void doPost (HttpServletRequest req,

HttpServletResponse res)throws IOException,
ServletException

```
{  
    PrintWriter out = res.getWriter();  
    int num1 =  
    Integer.parseInt(req.getParameter("num1")  
);    int num2 =  
    Integer.parseInt(req.getParameter("num2")  
);int tot = num1 + num2;  
    out.println("Your Total is " + tot);  
}  
  
}
```

TODO supply a title x +

localhost:8080/WebApplication1/

Gmail YouTube Maps MetaMask - Chrom...

Enter two numbers for addition...

Number 1:

Number 2:

Type here to search

GBP/INR -0.66% 10:14 AM 4/15/2023

localhost:8080/WebApplication1 x +

localhost:8080/WebApplication1/add

Gmail YouTube Maps MetaMask - Chrom...

Your Total is 7

Type here to search

GBP/INR -0.66% 10:15 AM 4/15/2023

9. Develop a JSP form to collect user registration details.

JSP Code:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>User Registration Results</title>

</head>

<body>

<h1>User Registration Results</h1>

<%

// Get form data

String username = request.getParameter("username");
String password = request.getParameter("password");
String email = request.getParameter("email");
String fullname = request.getParameter("fullname");
String dob = request.getParameter("dob");

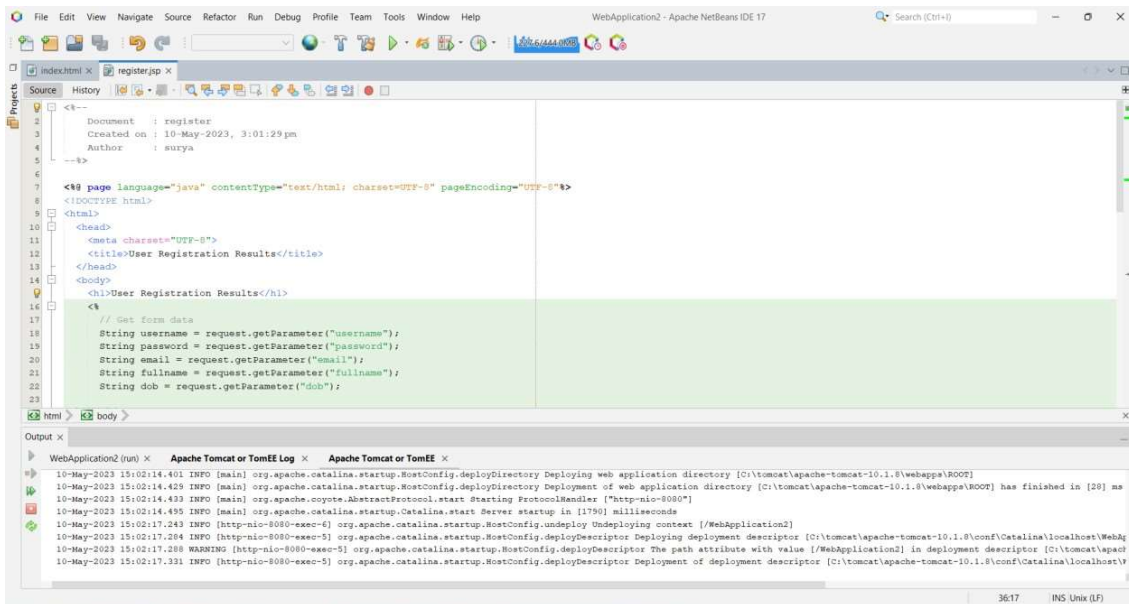
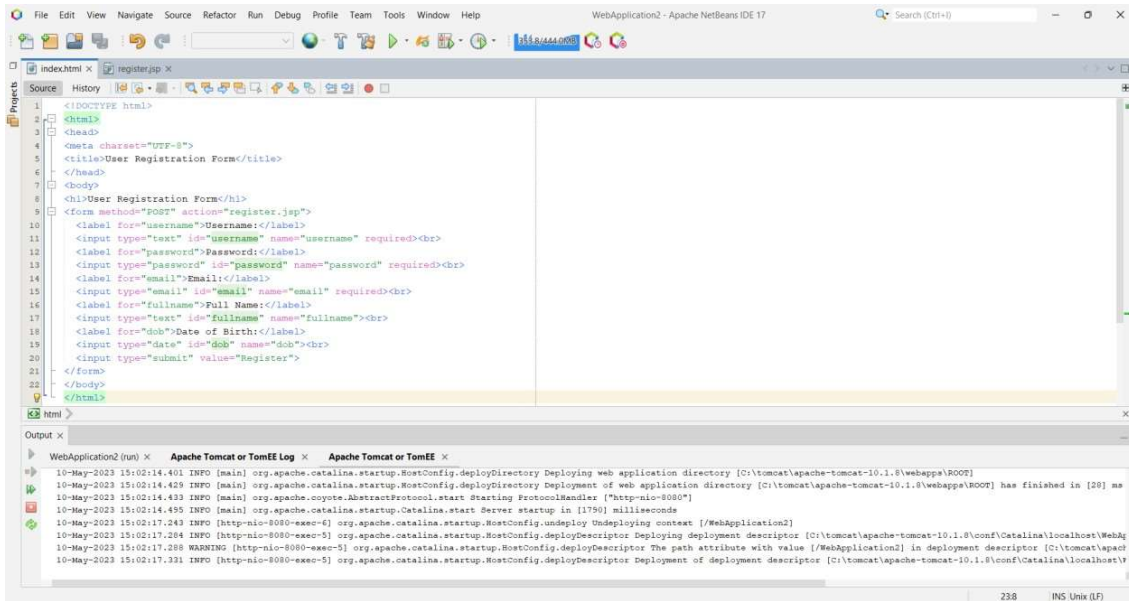
// Perform validation

boolean isValid = true;

if (username == null || username.isEmpty()) {
    isValid = false;
    out.println("<p>Username is required.</p>");
}

if (password == null || password.isEmpty()) {
    isValid = false;
    out.println("<p>Password is required.</p>");
```

```
}  
if (email == null || email.isEmpty()) {  
    isValid = false;  
    out.println("<p>Email is required.</p>");  
}  
  
// If data is valid, store in database  
if (isValid) {  
    // TODO: Store data in database or perform other actions  
    out.println("<p>Registration successful!</p>");  
    out.println("<p>Username: " + username + "</p>");  
    out.println("<p>Email: " + email + "</p>");  
    if (fullname != null && !fullname.isEmpty()) {  
        out.println("<p>Full Name: " + fullname + "</p>");  
    }  
    if (dob != null && !dob.isEmpty()) {  
        out.println("<p>Date of Birth: " + dob + "</p>");  
    }  
}  
%>  
</body>
```


User Registration Form

Username:

Password:

Email:

Full Name:

Date of Birth: 

User Registration Results

Registration successful!

Username: Surya_9122

Email: sspsurya2002@gmail.com

Full Name: Surya Prakash S

Date of Birth: 2002-12-09

10. Develop a JSP login form with cookies.

JSP Code:

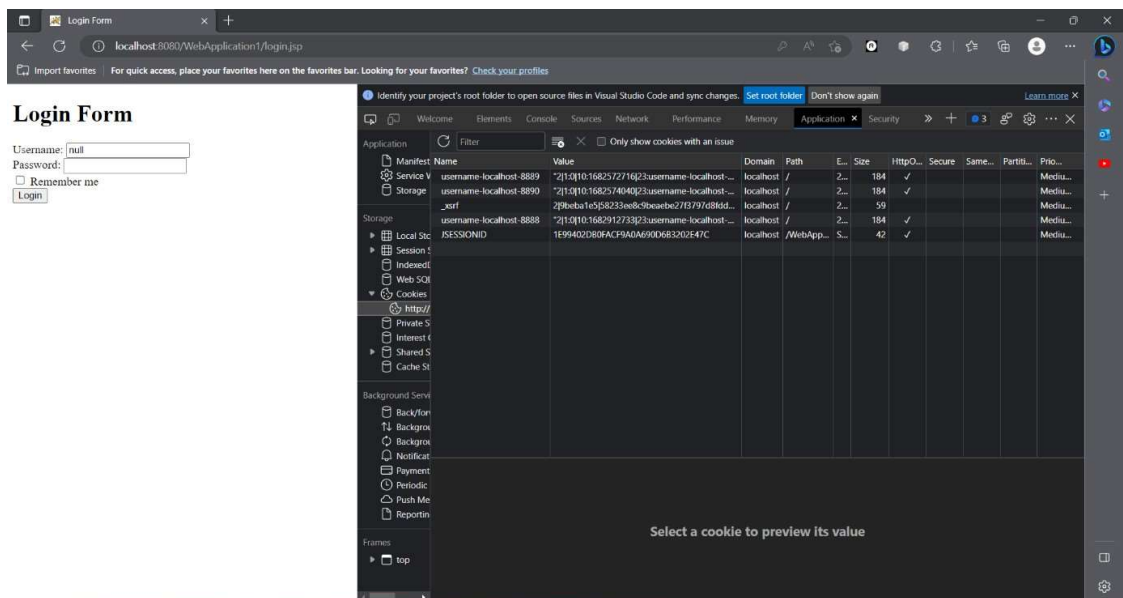
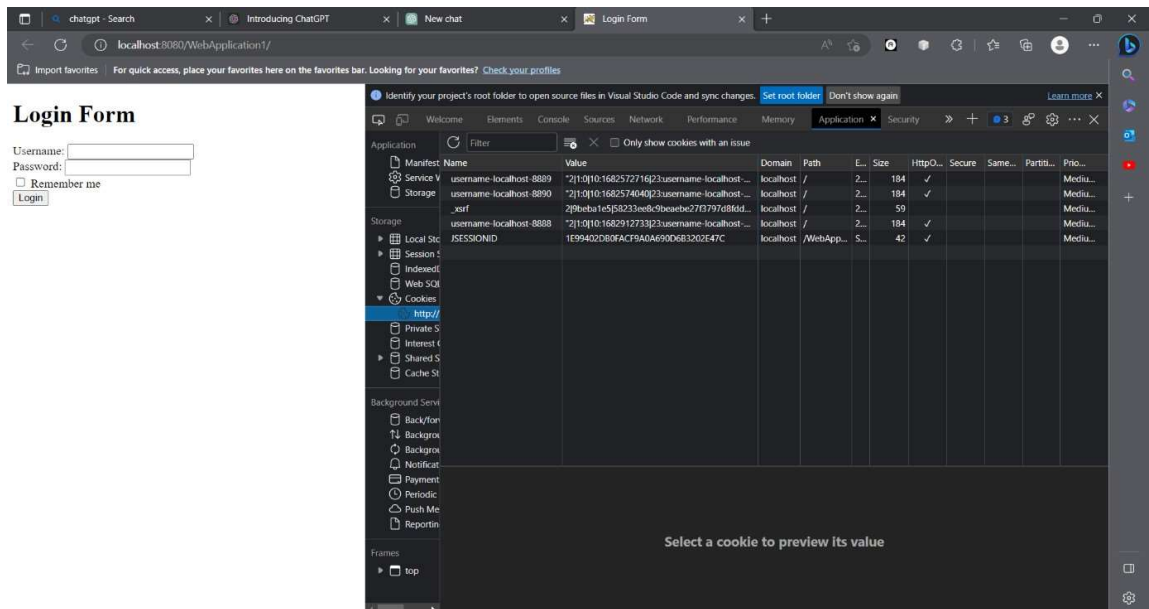
```
<%--
    Document : login
    Created on : 10-May-2023, 3:32:06 pm
    Author : surya
--%>

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Login Form - Processing</title>
</head>
<body>
<%
String username = request.getParameter("username");
String password = request.getParameter("password");
String remember = request.getParameter("remember");

// Check if the username and password are valid
if (isValidLogin(username, password)) {
    // If the user checked "Remember me", set a cookie for the username
    if (remember != null) {
        Cookie cookie = new Cookie("username", username);
        cookie.setMaxAge(60*60*24*30); // Cookie lasts for 30 days
        response.addCookie(cookie);
    }
    // Redirect to the main page for authenticated users
    response.sendRedirect("main.jsp");
} else {
    // If the login is invalid, show an error message
    out.println("<h2>Invalid login</h2>");
}
%>
</body>
</html>
```

<%!

```
private boolean isValidLogin(String username, String password) {  
    // Implement the logic to validate the login credentials  
    // For example, check against a database or hardcoded values  
    // Return true if valid, false otherwise  
    return (username.equals("user") && password.equals("password"));  
}  
%>
```



The screenshot shows a web browser with the address bar displaying 'localhost:8080/WebApplication1/login.jsp'. The page title is 'Login Form'. The form contains fields for 'Username:' (with the value 'null') and 'Password:', a 'Remember me' checkbox, and a 'Login' button. Below the browser window, the Visual Studio Code interface is open, showing the 'Application' tab of the browser's developer tools. The 'Only show cookies with an issue' filter is selected. A table of cookies is displayed, with columns for Manifest, Name, Value, Domain, Path, Expires / ... S., H., Secure, Same..., Partit..., and Priority. The 'JSESSIONID' cookie is highlighted, and its value is shown in the 'Cookie Value' field below the table. The value is '2110116257404923username-localhost-8890441m10J0DUCDCWtNBNGV02GyNZM5YUwZmnhNGQzZmV...jPa5781b0d6defe107e3ee5193c295b25a530d432e1045ab65c4d4910094adf'.

11. Apply JavaBean class to print information about a student class.

Student.class:

```
package com.mycompany.studentinfoprinter;
```

```
/*
```

```
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-  
default.txt to change this license
```

```
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit  
this template
```

```
*/
```

```
/**
```

```
 *
```

```
 * @author surya
```

```
*/
```

```
public class Student {
```

```
    private String name;
```

```
    private int age;
```

```
    private String major;
```

```
    public Student() {
```

```
        // default constructor
```

```
    }
```

```
    public Student(String name, int age, String major) {
```

```
        this.name = name;
```

```
        this.age = age;
```

```
        this.major = major;
```

```
    }
```

```
    // getters and setters for the private fields
```

```
    public String getName() {
```

```
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getMajor() {
        return major;
    }

    public void setMajor(String major) {
        this.major = major;
    }
}
```

StudentInfoPrinter.class:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 * to change this license
 */

package com.mycompany.studentinfoprinter;

/**
 *
 * @author surya
 */
```

```
*/  
  
public class StudentInfoPrinter {  
    public static void main(String[] args) {  
        // create a new student object using the default constructor  
        Student student1 = new Student();  
        student1.setName("John Smith");  
        student1.setAge(20);  
        student1.setMajor("Computer Science");  
  
        // create a new student object using the parameterized constructor  
        Student student2 = new Student("Jane Doe", 22, "Biology");  
  
        // print the information about the students  
        System.out.println("Student 1:");  
        System.out.println("Name: " + student1.getName());  
        System.out.println("Age: " + student1.getAge());  
        System.out.println("Major: " + student1.getMajor());  
        System.out.println();  
  
        System.out.println("Student 2:");  
        System.out.println("Name: " + student2.getName());  
        System.out.println("Age: " + student2.getAge());  
        System.out.println("Major: " + student2.getMajor());  
    }  
}
```


StudentInfoPrinter - Apache NetBeans IDE 17

```
package com.mycompany.studentinfoprinter;

/**
 *
 * @author surya
 */
public class StudentInfoPrinter {
    public static void main(String[] args) {
        // create a new student object using the default constructor
        Student student1 = new Student();
        student1.setName(name: "John Smith");
        student1.setAge(age: 20);
        student1.setMajor(major: "Computer Science");

        // create a new student object using the parameterized constructor
        Student student2 = new Student(name: "Jane Doe", age: 22, major: "Biology");

        // print the information about the students
        System.out.println("Student 1:");
        System.out.println("Name: " + student1.getName());
        System.out.println("Age: " + student1.getAge());
        System.out.println("Major: " + student1.getMajor());
        System.out.println();

        System.out.println("Student 2:");
        System.out.println("Name: " + student2.getName());
        System.out.println("Age: " + student2.getAge());
        System.out.println("Major: " + student2.getMajor());
    }
}
```

com.mycompany.studentinfoprinter.StudentInfoPrinter > main

Output x

WebApplication1 (run) x Apache Tomcat or TomEE Log x Apache Tomcat or TomEE x Run (StudentInfoPrinter) x

12:43 INS Unix (LF)

StudentInfoPrinter - Apache NetBeans IDE 17

```
package com.mycompany.studentinfoprinter;

/**
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
public class Student {
    private String name;
    private int age;
    private String major;
    public Student() {
        // default constructor
    }
    public Student(String name, int age, String major) {
        this.name = name;
        this.age = age;
        this.major = major;
    }
    // getters and setters for the private fields
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public String getMajor() {
        return major;
    }
    public void setMajor(String major) {
        this.major = major;
    }
}
```

com.mycompany.studentinfoprinter.Student > Student

Output x

WebApplication1 (run) x Apache Tomcat or TomEE Log x Apache Tomcat or TomEE x Run (StudentInfoPrinter) x

15:27 INS Windows (CRLF)

StudentInfoPrinter - Apache NetBeans IDE 17

```
package com.mycompany.studentinfoprinter;

/**
 *
 * @author surya
 */
public class StudentInfoPrinter {
    public static void main(String[] args) {
        // create a new student object using the default constructor
    }
}
```

com.mycompany.studentinfoprinter.StudentInfoPrinter > main

Output x

WebApplication1 (run) x Apache Tomcat or TomEE Log x Apache Tomcat or TomEE x Run (StudentInfoPrinter) x

Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be used instead of their jar artifacts.

Scanning for projects...

----- com.mycompany:studentinfoprinter -----

Building studentinfoprinter 1.0-EMAS800

[jar]

----- exec-maven-plugin:1.3.1:exec (default-cli) @ studentinfoprinter -----

Student 1:
Name: John Smith
Age: 20
Major: Computer Science

Student 2:
Name: Jane Doe
Age: 22
Major: Biology

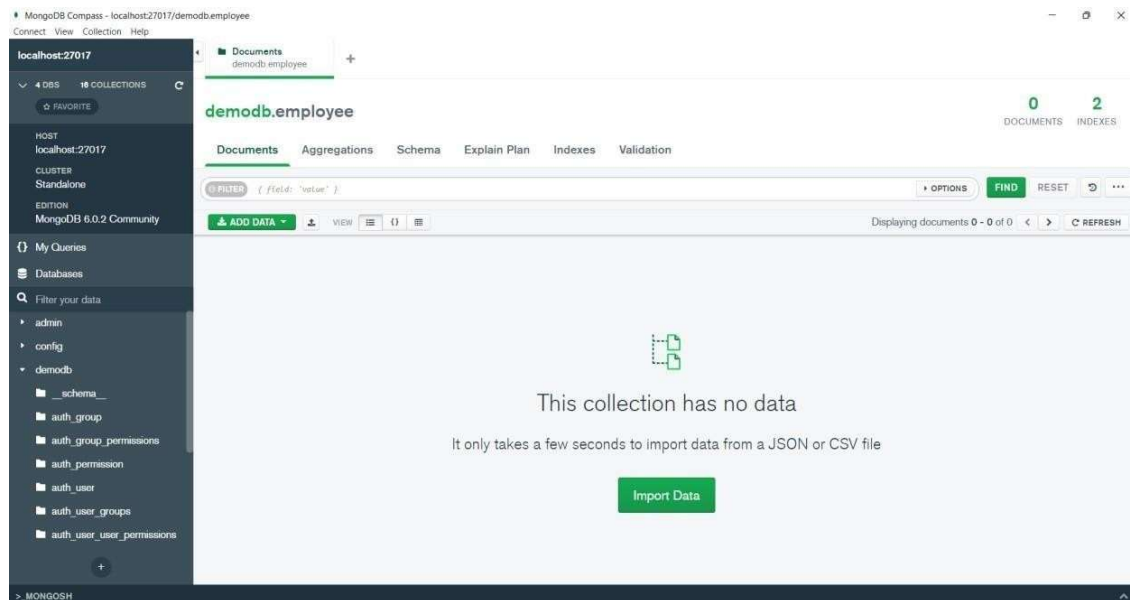
----- BUILD SUCCESS -----

Total time: 0.409 s
Finished at: 2023-05-11T06:50:52+05:30

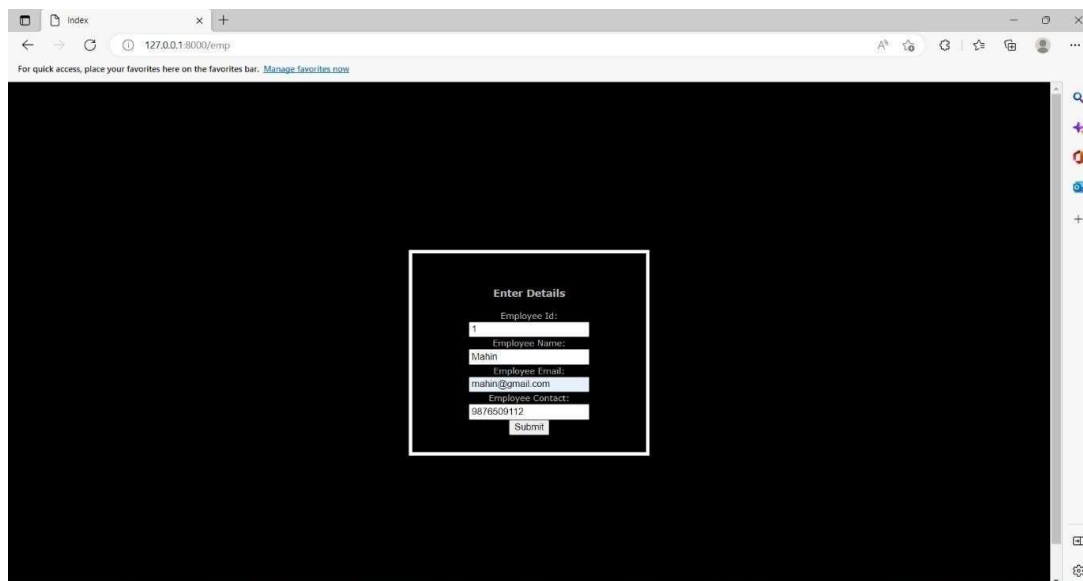
12:43 INS Unix (LF)

12. For Front end webpage connect with backend database and perform simple CRUD operations using JSF, PHP etc.

Empty Database:



Inserting data



Index x +

127.0.0.1:8000/emp

For quick access, place your favorites here on the favorites bar. [Manage favorites now](#)

Enter Details

Employee Id:

Employee Name:

Employee Email:

Employee Contact:

Index x +

127.0.0.1:8000/emp

For quick access, place your favorites here on the favorites bar. [Manage favorites now](#)

Enter Details

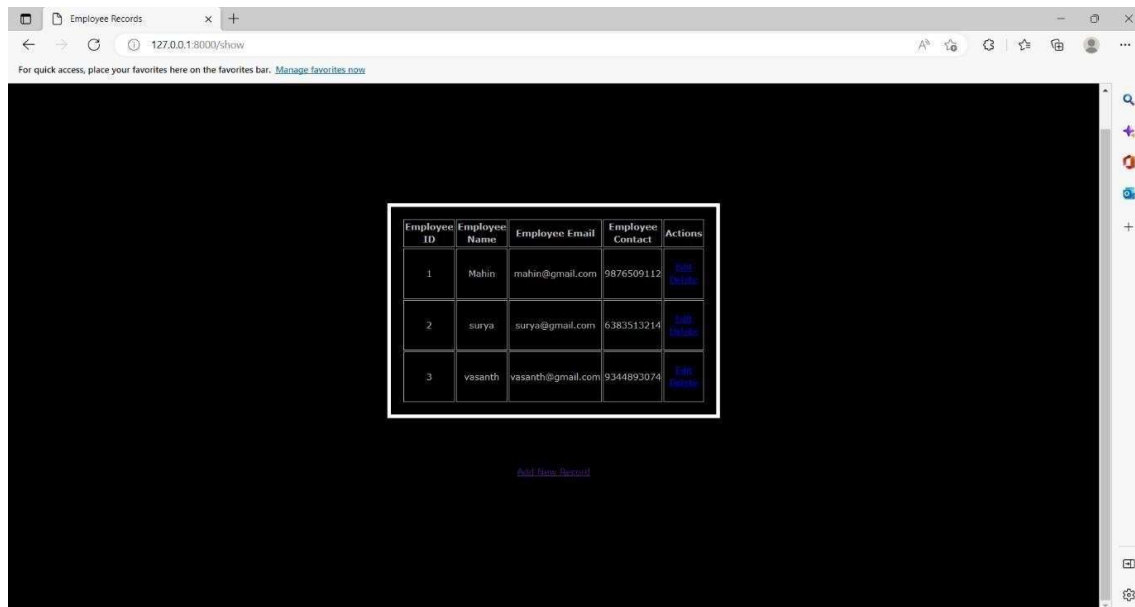
Employee Id:

Employee Name:

Employee Email:

Employee Contact:

Reading Data

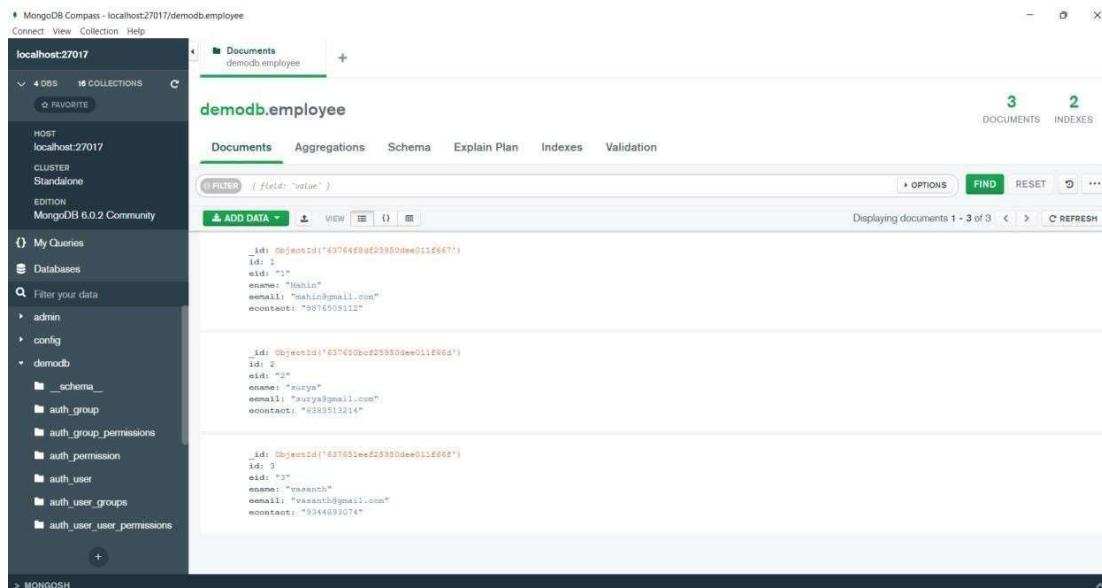


The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/show'. The page content features a table with five columns: Employee ID, Employee Name, Employee Email, Employee Contact, and Actions. There are three rows of data. Below the table is a link that says 'Add New Record'.

Employee ID	Employee Name	Employee Email	Employee Contact	Actions
1	Mahin	mahin@gmail.com	9876509112	Add Delete
2	surya	surya@gmail.com	6383513214	Add Delete
3	vasanth	vasanth@gmail.com	9344893074	Add Delete

[Add New Record](#)

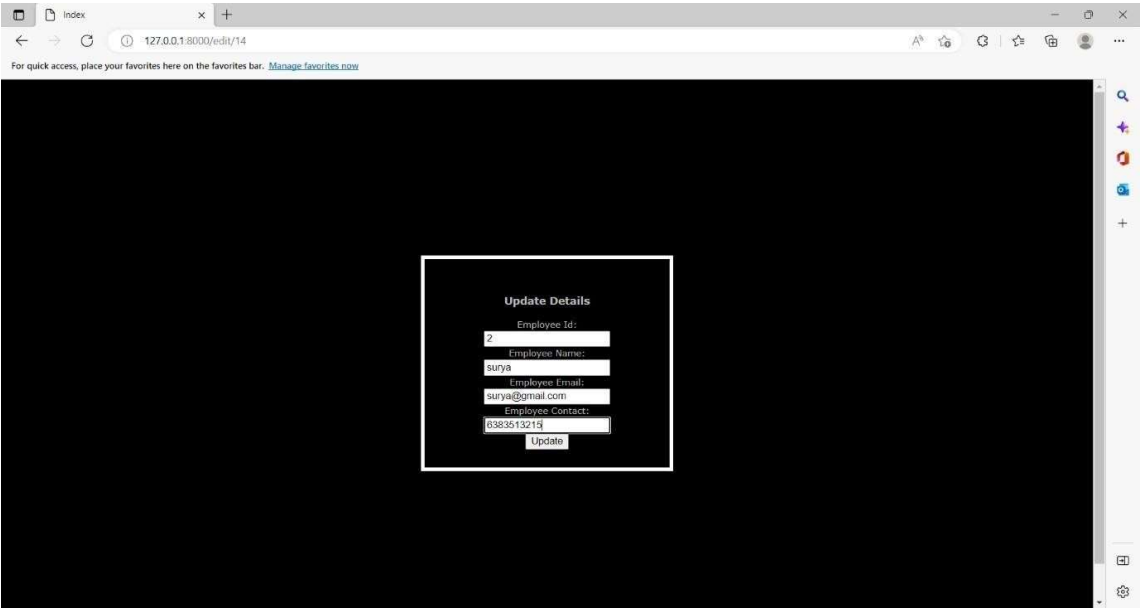
Reading Data in Database after Inserting



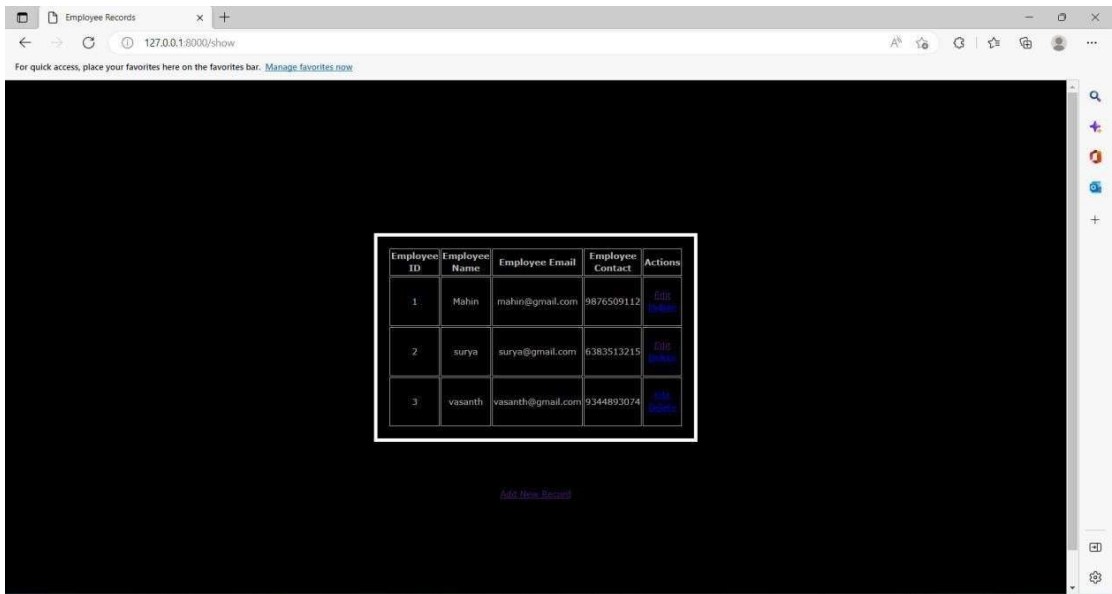
The screenshot shows the MongoDB Compass interface. The left sidebar shows the database structure with 'demodb' selected. The main panel displays the 'demodb.employee' collection with 3 documents. The documents are shown in a JSON format.

```
{ "_id": ObjectId("637650911200000000000001"), "id": 1, "name": "Mahin", "email": "mahin@gmail.com", "contact": "9876509112" }, { "_id": ObjectId("637650911200000000000002"), "id": 2, "name": "surya", "email": "surya@gmail.com", "contact": "6383513214" }, { "_id": ObjectId("637650911200000000000003"), "id": 3, "name": "vasanth", "email": "vasanth@gmail.com", "contact": "9344893074" }
```

Update Data



Reading data after updating



Deleting Data

