1. a) Design an HTML page having the following field text box, list, buttons, checkbox,radio, button, text area, select.

b) Validate the email id field by JavaScript.

c) Validate password filed by alpha-numeric.

Code:

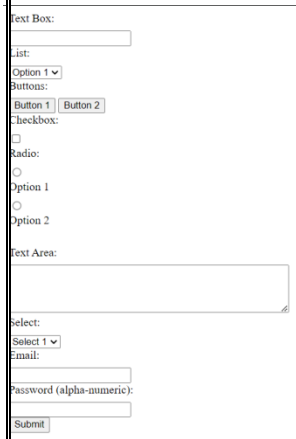
```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Form Validation</title>
 <style>
   label {
     display: block;
     margin-bottom: 8px;
   }
 </style>
</head>
<body>

 <form id="myForm" onsubmit="return validateForm()">
   <label for="username">Text Box:</label>
   <input type="text" id="username" name="username" required>

   <label for="myList">List:</label>
   <select id="myList" name="myList" required>
     <option value="option1">Option 1</option>
     <option value="option2">Option 2</option>
     <option value="option3">Option 3</option>
   </select>
```

```
<label>Buttons:</label>
<button type="button">Button 1</button>
<button type="button">Button 2</button>

<br>

<label for="myCheckbox">Checkbox:</label>
<input type="checkbox" id="myCheckbox" name="myCheckbox">

<label>Radio:</label>
<input type="radio" id="radio1" name="myRadio" value="option1">
<label for="radio1">Option 1</label>
<input type="radio" id="radio2" name="myRadio" value="option2">
<label for="radio2">Option 2</label>

<br>

<label for="myTextArea">Text Area:</label>
<textarea id="myTextArea" name="myTextArea" rows="4" cols="50" required></textarea>

<label for="mySelect">Select:</label>
<select id="mySelect" name="mySelect" required>
  <option value="select1">Select 1</option>
  <option value="select2">Select 2</option>
  <option value="select3">Select 3</option>
</select>

<br>

<label for="email">Email:</label>
```

```
  <input type="email" id="email" name="email" required>

  <label for="password">Password (alpha-numeric):</label>
  <input type="password" id="password" name="password" pattern="^(?=.[A-Za-z])(?=.\d)[A-Za-z\d]{8,}$" title="Must contain at least one letter and one number, and at least 8 or more characters">

  <br>

  <input type="submit" value="Submit">
 </form>

 <script>
  function validateForm() {
   // Email validation
   var emailInput = document.getElementById('email');
   var emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
   if (!emailRegex.test(emailInput.value)) {
    alert('Invalid email address');
    return false;
   }

   // Password validation
   var passwordInput = document.getElementById('password');
   var passwordRegex = /^(?=.[A-Za-z])(?=.\d)[A-Za-z\d]{8,}$/;
   if (!passwordRegex.test(passwordInput.value)) {
    alert('Invalid password. It must contain at least one letter and one number, and at least 8 or more characters');
    return false;
   }

   // Continue with form submission if all validations pass
   return true;
```

```
  }

 </script>


</body>

</html>
```

Output :



4. Design Following Web pages by <framesets> and <frame> tags

Code :

```
<!DOCTYPE html>

<html>

<frameset cols="*,*">

   <frame src="./frame_1.html">

   <frame src="./frame_2.html">


</frameset>

</html>
```

frame1.html

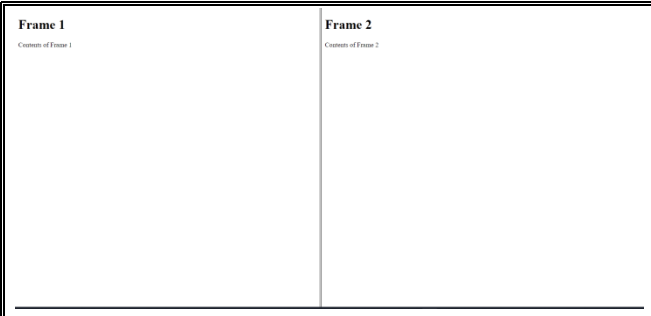
```
<!DOCTYPE html>

<html lang="en">

<head>
```

```
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

    <h1>Frame 1</h1>
    <p>Contents of Frame 1</p>

</body>
</html>
```

frame2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

    <h1>Frame 2</h1>
    <p>Contents of Frame 2</p>

</body>
</html>
```

Output :




```
<!DOCTYPE html>
<html>
<frameset rows="*,*">
   <frame src="./frame_1.html">
   <frame src="./frame_2.html">

</frameset>
</html>
```

frame1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>

      <h1>Frame 1</h1>
      <p>Contents of Frame 1</p>

</body>
</html>
```

frame2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

    <h1>Frame 2</h1>
    <p>Contents of Frame 2</p>

</body>
</html>
```

Output :


```
<!DOCTYPE html>
<html>

    <frameset rows=",">
      <frame src="./frame_1.html">
        <frameset cols=",">
          <frame src="./frame_1.html">
            <frame src="./frame_2.html">
        </frameset>

</frameset>
```

```
</html>
```

frame1.html

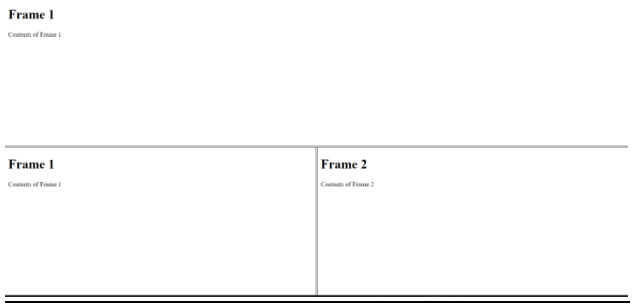
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

    <h1>Frame 1</h1>
    <p>Contents of Frame 1</p>

</body>
</html>
```

frame2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

    <h1>Frame 2</h1>
    <p>Contents of Frame 2</p>

</body>
```

```
</html>
```

Output :



7 . Develop an Job Portal where an user can search job skill and location wise and then apply the job. After applying job user can check the status.

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <link rel="stylesheet" href="style.css">
 <title>Job Portal</title>
</head>
<body>
 <h1>Job Portal</h1>
 <label for="skills">Skills:</label>
 <input type="text" id="skills" placeholder="Enter skills">
 <label for="location">Location:</label>
 <input type="text" id="location" placeholder="Enter location">
 <button onclick="searchJobs()">Search Jobs</button>

 <div id="job-listings"></div>

 <script src="script.js"></script>
</body>
</html>
```

style.css

```
body {
   font-family: Arial, sans-serif;
 }

 h1 {
   text-align: center;
```

```
}

label {
  display: block;
  margin-top: 10px;
}

input {
  width: 100%;
  padding: 8px;
  margin-top: 5px;
}

button {
  margin-top: 10px;
  padding: 8px;
  cursor: pointer;
}

#job-listings {
  margin-top: 20px;
}
```

script.js

```
function searchJobs() {
  const skills = document.getElementById('skills').value.split(',').map(skill => skill.trim());
  const location = document.getElementById('location').value.trim();

  fetch('/api/search-jobs', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({ skills, location }),
  })
    .then(response => response.json())
    .then(jobListings => displayJobListings(jobListings))
    .catch(error => console.error('Error:', error));
}

function displayJobListings(jobListings) {
  const jobListingsDiv = document.getElementById('job-listings');
  jobListingsDiv.innerHTML = '';
```

```
  if (jobListings.length === 0) {
    jobListingsDiv.innerHTML = '<p>No jobs found</p>';
    return;
  }

  jobListings.forEach(job => {
    const jobDiv = document.createElement('div');
    jobDiv.innerHTML = `<h3>${job.title}</h3>
                        <p>Skills: ${job.skills.join(', ')}</p>
                        <p>Location: ${job.location}</p>
                        <button onclick="applyJob(${job.id})">Apply Now</button>`;
    jobListingsDiv.appendChild(jobDiv);
  });
}

function applyJob(jobId) {
  const userId = 1; // Assume a user is logged in with ID 1 (for simplicity)

  fetch('/api/apply-job', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({ jobId, userId }),
  })
    .then(response => response.json())
    .then(data => alert(data.message))
    .catch(error => console.error('Error:', error));
}
```

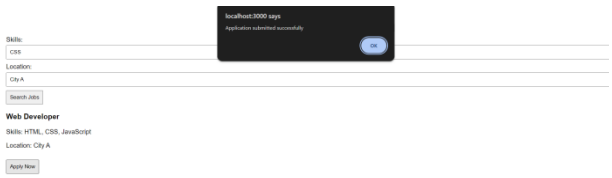
server.js

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

const bodyParser = require('body-parser');

const app = express();

const PORT = 3000;


// Dummy data for job listings and applications

let jobListings = [

  { id: 1, title: 'Web Developer', skills: ['HTML', 'CSS', 'JavaScript'], location: 'City A' },
```

```
  { id: 2, title: 'Data Scientist', skills: ['Python', 'Machine Learning'], location: 'City B' },
];

let jobApplications = [];

app.use(bodyParser.json());
app.use(express.static('public'));

// Endpoint to get job listings based on skills and location
app.post('/api/search-jobs', (req, res) => {
  const { skills, location } = req.body;
  const filteredJobs = jobListings.filter(job =>
    job.skills.some(skill => skills.includes(skill)) && job.location === location
  );
  res.json(filteredJobs);
});

// Endpoint to apply for a job
app.post('/api/apply-job', (req, res) => {
  const { jobId, userId } = req.body;
  const application = { jobId, userId, status: 'Applied' };
  jobApplications.push(application);
  res.json({ message: 'Application submitted successfully' });
});

// Endpoint to check application status
app.get('/api/application-status/:userId', (req, res) => {
  const userId = parseInt(req.params.userId);
  const userApplications = jobApplications.filter(app => app.userId === userId);
  res.json(userApplications);
});
```

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

  console.log(`Server is running on http://localhost:${PORT}`);

});
```

Output :



10 . b Design web pages with following CSS

a)External

b)Embedded

c)Inline

d)Imported

a) External :

index.html

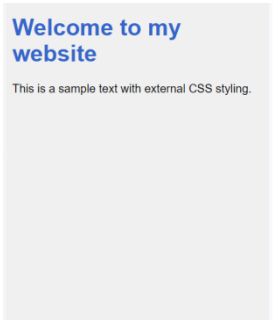
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="styles.css">
   <title>External CSS Example</title>
</head>
<body>
   <h1 class="heading">Welcome to my website</h1>
   <p class="content">This is a sample text with external CSS styling.</p>
</body>
</html>
```

style.css

```
/* External CSS */
body {
   background-color: #f0f0f0;
   font-family: Arial, sans-serif;
}

.heading {
   color: #3366cc;
}
```

```
.content {
   font-size: 16px;
}
```

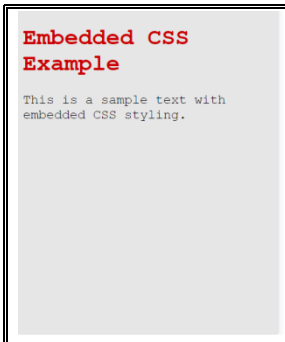
Output :



b) Embedded :

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Embedded CSS Example</title>
   <style>
      /* Embedded CSS */
      body {
         background-color: #e6e6e6;
         font-family: 'Courier New', Courier, monospace;
      }

      .heading {
         color: #cc0000;
      }

      .content {
         font-size: 18px;
      }
   </style>
</head>
<body>
   <h1 class="heading">Embedded CSS Example</h1>
   <p class="content">This is a sample text with embedded CSS styling.</p>
</body>
</html>
```

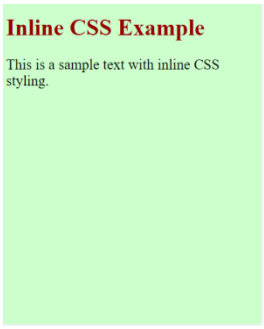
Output:



c) Inline

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Inline CSS Example</title>
</head>
<body style="background-color: #ccffcc; font-family: 'Times New Roman', Times, serif;">
   <h1 style="color: #990000;">Inline CSS Example</h1>
   <p style="font-size: 20px;">This is a sample text with inline CSS styling.</p>
</body>
</html>
```

Output:



d) Imported

main.css

```
/* Imported CSS */
body {
   margin: 0;
   padding: 0;
   font-family: 'Verdana', sans-serif;
}

.container {
   width: 80%;
   margin: 0 auto;
}

.header {
   background-color: #333;
   color: #fff;
   padding: 10px;
}
```

```
.main-content {
  padding: 20px;
}
```

style.css

```
/* Imported CSS */
@import url('main.css');
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="styles.css">
  <title>Imported CSS Example</title>
</head>
<body>
  <div class="container">
    <header class="header">
      <h1>Imported CSS Example</h1>
    </header>
    <div class="main-content">
      <p>This is a sample text with imported CSS styling.</p>
    </div>
  </div>
</body>
</html>
```

Output :

8. Develop a web application that will display a whole table stored in database.

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>Database Records</title>
```

```
  <style>
    table{
      width: 70%;
      margin: auto;
      font-family: Arial, Helvetica, sans-serif;
    }
    table, tr, th, td{
      border: 1px solid #d4d4d4;
      border-collapse: collapse;
      padding: 12px;
    }
    th, td{
      text-align: left;
      vertical-align: top;
    }
    tr:nth-child(even){
      background-color: #e7e9eb;
    }
  </style>
<body>

<?php
  //storing database details in variables.
  $hostname = "localhost";
  $username = "webcodzingdb_user";
  $password = "webcodzing123";
  $dbname = "webcodzing_db";

  //creating connection to database
  $con = mysqli_connect($hostname, $username, $password, $dbname);
  //checking if connection is working or not
  if(!$con)
  {
    die("Connection failed!" . mysqli_connect_error());
  }
  else
  {
    echo "Successfully Connected! <br>";
  }
  //Output Form Entries from the Database
  $sql = "SELECT id, name_fld, email_fld, msg_fld FROM contactform_entries";
  //fire query
```

```
  $result = mysqli_query($con, $sql);
  if(mysqli_num_rows($result) > 0)
  {
    echo '<table> <tr> <th> Id </th> <th> Name </th> <th> Email </th> <th> Message </th> </tr>';
    while($row = mysqli_fetch_assoc($result)){
     // to output mysql data in HTML table format
       echo '<tr > <td>' . $row["id"] . '</td>
       <td>' . $row["name_fld"] . '</td>
       <td> ' . $row["email_fld"] . '</td>
       <td>' . $row["msg_fld"] . '</td> </tr>';
    }
    echo '</table>';
  }
  else
  {
     echo "0 results";
  }
  // closing connection
  mysqli_close($con);

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

Output:

Successfully Connected!

Id	Name	Email	Message
19	John	john123@gmail.com	Hi, I am John. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.
20	David	david123@gmail.com	Hi, I am David. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.
21	Christine	christine123@gmail.c	Hi, I am Christine. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.
22	Chris Evans	chris123@gmail.com	Hi, I am Chris. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.
23	Jack	jack123@gmail.com	Hi, I am Jack. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.
24	Nick	nick123@gmail.com	Hi, I am Nick. Lorem ipsum dolor sit, amet consectetur adipisicing elit. Autem quia a aut illo praesentium consequuntur tempore molitia et, asperiores, ducimus, dolores modi quaerat quibusdam nam quis ullam. Repudiandae, veniam nobis.

9. Develop a web application where user name, user id, principle amount ,rate of interest and time will be provided by the HTML interface and calculate the simple interest.

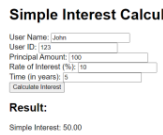
Code :

```
<!DOCTYPE html>

<html lang="en">

<head>

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

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

<title>Simple Interest Calculator</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

</style>

</head>

<body>

<h1>Simple Interest Calculator</h1>

<form id="interestForm">

<label for="userName">User Name:</label>

<input type="text" id="userName" required><br>



<label for="userId">User ID:</label>

<input type="text" id="userId" required><br>



<label for="principal">Principal Amount:</label>

<input type="number" id="principal" required><br>



<label for="rate">Rate of Interest (%):</label>

<input type="number" id="rate" required><br>



<label for="time">Time (in years):</label>

<input type="number" id="time" required><br>
```

```
<button type="button" onclick="calculateInterest()">Calculate Interest</button>

</form>



<h2>Result:</h2>

<p id="result"></p>



<script>

function calculateInterest() {

// Get input values

const principal = parseFloat(document.getElementById('principal').value);

const rate = parseFloat(document.getElementById('rate').value);

const time = parseFloat(document.getElementById('time').value);



// Calculate simple interest

const interest = (principal * rate * time) / 100;



// Display the result

document.getElementById('result').innerHTML = `Simple Interest: ${interest.toFixed(2)}`;

}

</script>

</body>

</html>
```

Output :



6. Develop a web application where user name, password will be provided by the HTML interface and check the login authentication and password filed with the value stored in database.

Code :

form.html

```
<html>

<head>

<title>PHP login system</title>

<link rel = "stylesheet" type = "text/css" href = "style.css">

</head>

<body>

<div id = "frm">

<h1>Login</h1>

<form name="f1" action = "authentication.php" onsubmit = "return validation()" method = "POST">

<p>

<label> UserName: </label>

<input type = "text" id ="user" name = "user" />

</p>

<p>

<label> Password: </label>

<input type = "password" id ="pass" name = "pass" />

</p>
```

```
<p>
<input type = "submit" id = "btn" value = "Login" />
</p>
</form>
</div>
<script>
function validation()
{
var id=document.f1.user.value;
var ps=document.f1.pass.value;
if(id.length=="" && ps.length=="") {
alert("User Name and Password fields are empty");
return false;
}
else
{
if(id.length=="") {
alert("User Name is empty");
return false;
}
if (ps.length=="") {
alert("Password field is empty");
return false;
}
}
}
```

```
</script>

</body>

</html>
```

style.css

```
body{

background: #eee;

}

#frm{

border: solid gray 1px;

width:25%;

border-radius: 2px;

margin: 120px auto;

background: white;

padding: 50px;

}

#btn{

color: #fff;

background: #337ab7;

padding: 7px;

margin-left: 70%;

}
```

connection.php

```
<?php

$host = "localhost";

$user = "root";

$password = "";
```

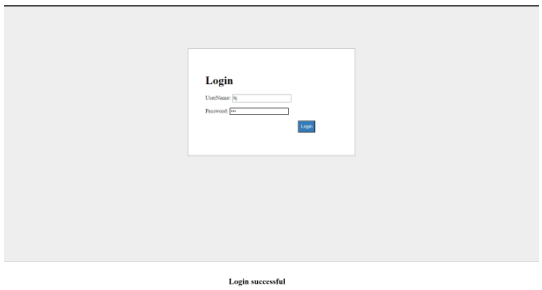
```
$db_name = "javatpoint";

$con = mysqli_connect($host, $user, $password, $db_name);

if(mysqli_connect_errno()) {

die("Failed to connect with MySQL: ". mysqli_connect_error());

}

?>
```

authentication.php

```
<?php

include('connection.php');

$username = $_POST['user'];

$password = $_POST['pass'];

//to prevent from mysqli injection

$username = stripcslashes($username);

$password = stripcslashes($password);

$username = mysqli_real_escape_string($con, $username);

$password = mysqli_real_escape_string($con, $password);

$sql = "select *from login where username = '$username' and password = '$password'";

$result = mysqli_query($con, $sql);

$row = mysqli_fetch_array($result, MYSQLI_ASSOC);

$count = mysqli_num_rows($result);

if($count == 1){

echo "<h1><center> Login successful </center></h1>";

}

else{

echo "<h1> Login failed. Invalid username or password.</h1>";

}
```

?>

Output:



5. Develop a web application that will store values in session object and retrieve the values from session object.

Code :

app.js

```
const express = require('express');
const session = require('express-session');

const app = express();

// Configure session middleware
app.use(session({
  secret: 'your-secret-key',
  resave: false,
  saveUninitialized: true
}));

// Serve HTML file
app.get('/', (req, res) => {
  res.sendFile(__dirname + '/index.html');
});

// Set session value
app.get('/set/:value', (req, res) => {
  req.session.value = req.params.value;
  res.send('Value set in session: ' + req.params.value);
});

// Get session value
```

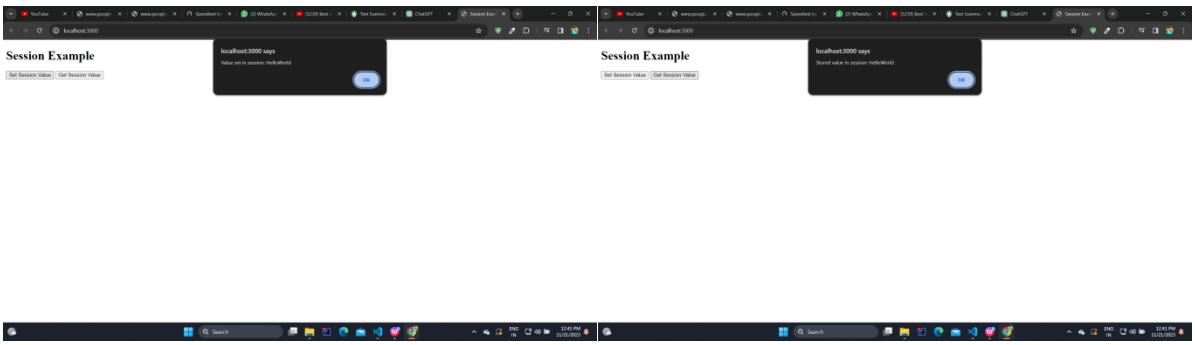
```
app.get('/get', (req, res) => {
  const storedValue = req.session.value || 'No value stored in session';
  res.send('Stored value in session: ' + storedValue);
});

const PORT = 3000;
app.listen(PORT, () => {
  console.log(`Server is running at http://localhost:${PORT}`);
});
```

index.js

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Session Example</title>
</head>
<body>
  <h1>Session Example</h1>
  <button onclick="setValue()">Set Session Value</button>
  <button onclick="getValue()">Get Session Value</button>

  <script>
    function setValue() {
      fetch('/set/HelloWorld')
        .then(response => response.text())
        .then(message => alert(message));
    }

    function getValue() {
```

```
    fetch('/get')
      .then(response => response.text())
      .then(message => alert(message));
  }
 </script>
</body>
</html>
```

Output :



10 a) Design an XML documents for Employee Element and where attributes will be name,gender,id,salary.Show the Schema and DTD also.

Code :

employee.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<employees>
  <employee id="101" name="John Doe" gender="Male" salary="50000" />
  <employee id="102" name="Jane Smith" gender="Female" salary="60000" />
  <!-- Add more employee elements as needed -->
</employees>
```

employee.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="employees">
    <xs:complexType>
```

```
        <xs:sequence>
          <xs:element name="employee" maxOccurs="unbounded">
            <xs:complexType>
              <xs:attribute name="id" type="xs:string" use="required"/>
              <xs:attribute name="name" type="xs:string" use="required"/>
              <xs:attribute name="gender" type="xs:string" use="required"/>
              <xs:attribute name="salary" type="xs:decimal" use="required"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>

</xs:schema>
```

employee.dtd

```
<!ELEMENT employees (employee+)>
<!ELEMENT employee EMPTY>
<!ATTLIST employee
  id     CDATA   #REQUIRED
  name    CDATA   #REQUIRED
  gender  CDATA   #REQUIRED
  salary  CDATA   #REQUIRED
>
```

3 a)Design an HTML page to create the following table

User Name	Password	Salary
A	123	10000
B	456	20000
C	789	30000

b) Design an HTML page to create hyper linking between two web pages.

b) Store the value of user name, user id and simple interest amount into the database.

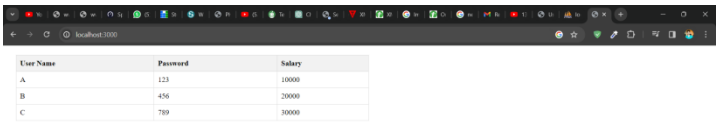
Code :

a)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>User Information Table</title>
  <style>
    table {
      border-collapse: collapse;
      width: 50%;
      margin: 20px;
    }

    th, td {
      border: 1px solid #ddd;
      padding: 8px;
      text-align: left;
    }

    th {
      background-color: #f2f2f2;
    }
  </style>
</head>
<body>
```

```
  <table>
    <thead>
      <tr>
        <th>User Name</th>
        <th>Password</th>
        <th>Salary</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>A</td>
        <td>123</td>
        <td>10000</td>
      </tr>
      <tr>
        <td>B</td>
        <td>456</td>
        <td>20000</td>
      </tr>
      <tr>
        <td>C</td>
        <td>789</td>
        <td>30000</td>
      </tr>
    </tbody>
  </table>

</body>
</html>
```

Output :



b)

page1.html

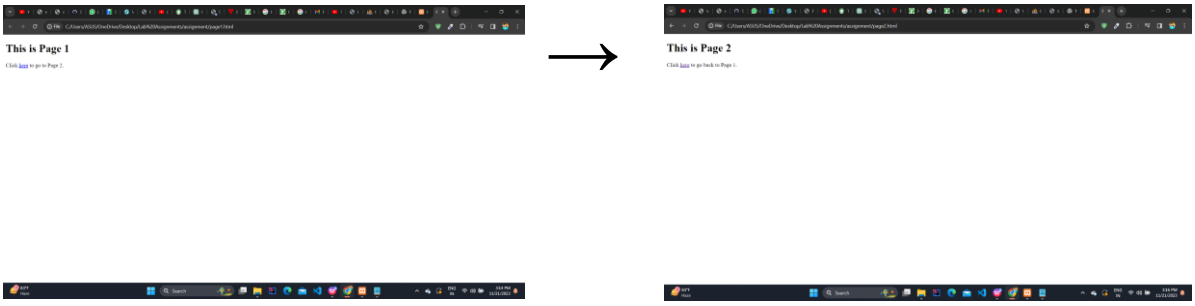
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Page 1</title>
</head>
<body>

   <h1>This is Page 1</h1>
   <p>Click <a href="page2.html">here</a> to go to Page 2.</p>

</body>
</html>
```

page2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Page 2</title>
</head>
<body>

    <h1>This is Page 2</h1>
    <p>Click <a href="page1.html">here</a> to go back to Page 1.</p>

</body>
</html>
```

Output:



app.js

```
const express = require('express');
const mysql = require('mysql');
const app = express();
const port = 3000;

// Create a connection to the MySQL database
const db = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: '',
    database: 'sample_db'
});
```

```
// Connect to the database
db.connect((err) => {
  if (err) {
    throw err;
  }
  console.log('Connected to MySQL database');
});

// Route for displaying the HTML page with the table
app.get('/', (req, res) => {
  res.sendFile(__dirname + '/index1.html');
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

Output :

2 . Implement a Java application
a) that will make connection with any database.
b) create table into that database
b) insert values into table.
c) retrieve all the values from table

Code :

a)

```
import java.sql.*;


public class DBConnect {
   public static void main(String[] args) {
      // Replace with your database URL, username, and password
      String url = "jdbc:mysql://localhost:3306/";
      String username = "root";
      String password = "";


      try {
         // Load the database driver
         Class.forName("com.mysql.cj.jdbc.Driver");


         // Create a connection to the database
         Connection conn = DriverManager.getConnection(url, username, password);


         System.out.println("Connected to the database successfully!");
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

b)

Code :

```
import java.sql.*;


public class CreateTable {
   public static void main(String[] args) {
      // Replace with your database URL, username, and password
      String url = "jdbc:mysql://localhost:3306/";
```

```
        String username = "root";
        String password = "";

        try {
            // Load the database driver
            Class.forName("com.mysql.cj.jdbc.Driver");

            // Create a connection to the database
            Connection conn = DriverManager.getConnection(url, username, password);

            // Create a table in the database
            String sql = "CREATE TABLE employees ("
                    + "id INTEGER not NULL AUTO_INCREMENT,"
                    + "name VARCHAR(255),"
                    + "position VARCHAR(255),"
                    + "salary FLOAT,"
                    + "PRIMARY KEY (id)"
                    + ")";
            Statement stmt = conn.createStatement();
            stmt.executeUpdate(sql);

            System.out.println("Table 'employees' created successfully!");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

c)

Code:

```
import java.sql.*;
```

```
public class InsertData {

   public static void main(String[] args) {

      // Replace with your database URL, username, and password

      String url = "jdbc:mysql://localhost:3306/";

      String username = "root";

      String password = "";


      try {

         // Load the database driver

         Class.forName("com.mysql.cj.jdbc.Driver");


         // Create a connection to the database

         Connection conn = DriverManager.getConnection(url, username, password);


         // Insert data into the table

         String sql = "INSERT INTO employees (name, position, salary) VALUES (?, ?, ?)";

         PreparedStatement pstmt = conn.prepareStatement(sql);

         pstmt.setString(1, "John Doe");

         pstmt.setString(2, "Software Engineer");

         pstmt.setFloat(3, 80000);

         pstmt.executeUpdate();


         System.out.println("Data inserted successfully!");

      } catch (Exception e) {

         e.printStackTrace();

      }

   }

}
```

d)

Code:

```
import java.sql.*;
```

```
public class RetrieveData {
   public static void main(String[] args) {
      // Replace with your database URL, username, and password
      String url = "jdbc:mysql://localhost:3306/";
      String username = "root";
      String password = "";

      try {
         // Load the database driver
         Class.forName("com.mysql.cj.jdbc.Driver");

         // Create a connection to the database
         Connection conn = DriverManager.getConnection(url, username, password);

         // Retrieve data from the table
         String sql = "SELECT * FROM employees";
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(sql);

         // Print the retrieved data
         while (rs.next()) {
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String position = rs.getString("position");
            float salary = rs.getFloat("salary");

            System.out.println("ID: " + id + ", Name: " + name + ", Position: " + position + ", Salary: " + salary);
         }
      } catch (Exception e) {
         e.printStackTrace();
```

```
    }

  }

}
```