\

Web Technologies

Verified by
Prof. D Sandeep

Prepared by
Mr. K Sandeep

1. Lab Instructions:

I. Follow all instructions carefully in the lab.

II. Without permission mobile phones are not allowed.

III. Formals, shoes and ID Cards are compulsory for the lab.

IV. Don't install or delete any software without permission.

V. Students should complete writing their experiment records before arriving at the laboratory and have them reviewed and signed by the Lab Instructor.

2. Lab Exam Instructions and Evaluation:

I. A total of 10 experiments, as outlined in the provided manual, collectively account for 100 marks.

II. There will be two separate examinations. The first examination will cover the first 5 experiments, and the second examination will assess the remaining 6 experiments. By default, these examinations will take place in the week following the completion of each set of 5 experiments.

III. During the examination, a single experiment will be randomly selected from these six for the exam evaluation.

IV. Each examination carries a total of 50 marks, distributed as follows:

3. Course Objectives: Upon successful completion, Students will have

- To understand Fundamental Web Technologies.
- To master Client-Side and Server-Side Scripting using JavaScript, PHP and JSP.
- To design and deploy applications using JQuery and AJAX.

4. Course Learning Outcomes:

CILO 1	Identify the difference between the HTML, CSS and XML documents and create web pages
CILO 2	Understand the concepts of JavaScript and PHP to create website.
CILO 3	Analyse and design web application which follows MVC architecture using jQuery, JSON and AJAX.

5. Module:

Module No.	Topics	CILO Mapping
Unit 1	Web basics, HTML, CSS	CILO 1
Unit 2	XML, Client-side scripting	CILO 1
Unit 3	PHP	CILO 2
Unit 4	JSP	CILO 3
Unit 5	jQuery, JSON, AJAX	CILO 4

6. Textbooks: Web Technologies, Uttam K Roy, Oxford University Press

The Complete Reference PHP — Steven Holzner, Tata McGraw-Hill.

Web Programming, building internet applications - Chris Bates, Wiley Dreamtech

7. Reference Books:

1Web Technologies, Uttam K Roy, Oxford University Press

2. The Complete Reference PHP — Steven Holzner, Tata McGraw-Hill.

3. Web Programming, building internet applications - Chris Bates, Wiley Dreamtech

4. Java Server Pages — Hans Bergsten, SPD O'Reilly

8. E-content:

1. HTML: HyperText Markup Language | MDN
2. <video>: The Video Embed element - HTML: HyperText Markup Language | MDN
3. CodePen: Online Code Editor and Front End Web Developer Community
4. Replit – Build apps and sites with AI
5. Home | Web Accessibility Initiative (WAI) | W3C
6. Website Builder - Create a Free Website Today | Wix.com
7. Responsive design - Learn web development | MDN
8. Shopify
9. SEO for everyone • Yoast
10. JavaScript Guide - JavaScript | MDN

9. Attendance Policy:

1. A Student must normally maintain a minimum of 75% attendance in the course.
2. Lab without which he/she shall be disqualified from appearing in the respective Lab examination.

Evaluation Scheme:

Lab Exam – I:

Maximum Marks: 50 Mark

Sr.no.	Marks of Distribution	Marks
1	Performing Experiment [Compiling (10 marks), Execution and Desired output (10 marks)]	20
2	Record (20), Day to day performance (20)	40
3	Programming skill	20
4	Viva	20

Lab Exam – II:

Maximum Marks: 50 Mark

Sr.no.	Marks of Distribution	Marks
1	Performing Experiment [Compiling (10 marks), Execution and Desired output (10 marks)]	20
2	Record (20), Day to day performance (20)	40
3	Programming skill	20
4	Viva	20

INDEX

9	Create Home Page with two sub-pages one for login and one for registration, validate it using Jquery	70-77	CILO 4
10	Create two dropdowns, the values of first dropdown are fetched from DB using PHP/JSP and onchnage of value load the content in another dropdown without refreshing/reloading the page (Hint: Use AJAX) - Ex: States List are fetched from DB and on change of State, load the city names of the repsective state in another dropdown	78-83	CILO 4

Experiment 1

1. **Aim:** Create your own CV using HTML only
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

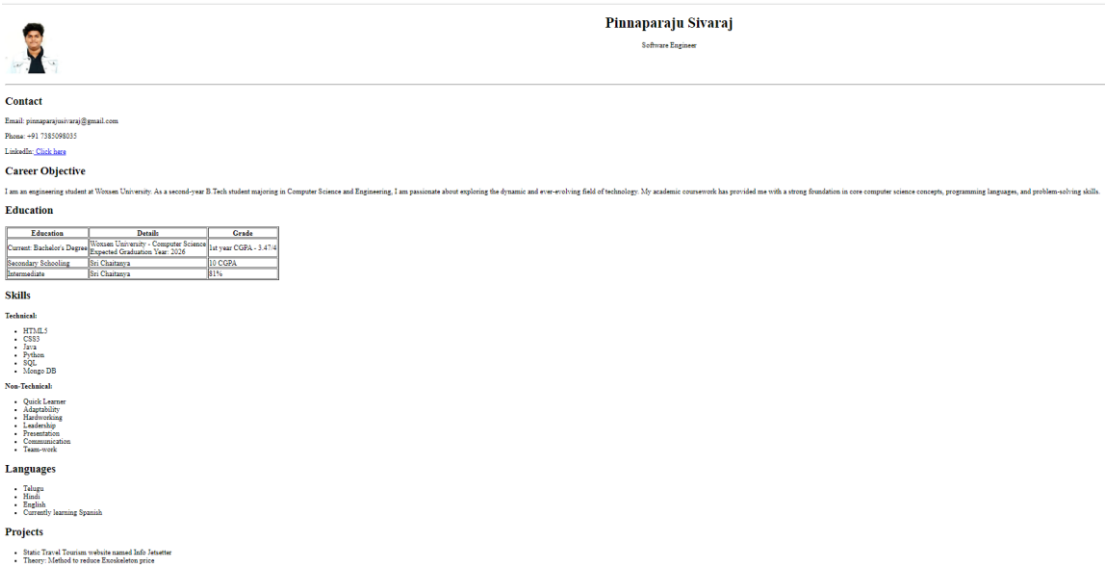
```
<!DOCTYPE html>
<html>
<head>
  <title> Sivaraj CV</title>
</head>
<body>
  <img src="https://media.licdn.com/dms/image/D4E03AQH7av2efH2AhA/profile-displayphoto-shrink_200_200/0/1673330337366?e=2147483647&v=beta&t=V84jZ9-1HxUhYiRXQtTA6RS_Ubwc0iOb4iNAv6NxSR8" style="  width: 130px; height: 150px alt = passport photo" align = "left"/>
  <h1 align = "center">Pinnaparaju Sivaraj</h1>
  <p align ="center">Software Engineer</p>

  <br><br><br>
  <hr>

  <h2>Contact</h2>
    <p>Email: pinnaparajusivaraj@gmail.com</p>
```

```
    <p>Phone: +91 7385098035</p>

    <p>LinkedIn:<a href="https://in.linkedin.com/in/pinnaparaju-sivaraj-06a489261">
Click here</a></p>


  <h2>Career Objective</h2>

  <p align ="left"> I am an engineering student at Woxsen University. As a second-year
B.Tech student majoring in Computer Science and Engineering, I am passionate about
exploring the dynamic and ever-evolving field of technology. My academic coursework has
provided me with a strong foundation in core computer science concepts, programming
languages, and problem-solving skills. </p>

  <h2>Education</h2>

  <table border="1">

    <tr>

      <th>Education</th>

      <th>Details</th>

      <th>Grade</th>

    </tr>

    <tr>

      <td>Current: Bachelor's Degree</td>

      <td>Woxsen University - Computer Science<br>Expected Graduation Year: 2026</td>

      <td>1st year CGPA - 3.47/4</td>

    </tr>

    <tr>

      <td>Secondary Schooling</td>

      <td>Sri Chaitanya</td>

      <td>10 CGPA</td>

    </tr>

    <tr>

      <td>Intermediate</td>
```

```
    <td>Sri Chaitanya</td>
    <td>81%</td>
  </tr>
</table>

    <h2>Skills</h2>
    <p><b>Technical:</b></p>
    <ul>
        <li>HTML5</li>
        <li>CSS3</li>
        <li>Java</li>
        <li>Python</li>
        <li>SQL</li>
        <li>Mongo DB</li>
    </ul>

    <p><b>Non-Technical:</b></p>
    <ul>
        <li>Quick Learner</li>
        <li>Adaptability</li>
        <li>Hardworking</li>
        <li>Leadership</li>
        <li>Presentation</li>
        <li>Communication</li>
        <li>Team-work</li>
    </ul>

    <h2>Languages</h2>
```

```
    <ul>
      <li>Telugu</li>
      <li>Hindi</li>
      <li>English</li>
      <li>Currently learning Spanish</li>
    </ul>

    <h2>Projects</h2>
    <ul>
      <li>Static Travel Tourism website named Info Jetsetter</li>
      <li>Theory: Method to reduce Exoskeleton price</li>
    </ul>

</body>
    </html>
```

5. **Observations:** It seems like you're asking for a more concise version of your HTML CV. Here's a more simplified and compact version of your code, with improvements for readability, structure, and better practices:
6. **Result:**



Pinnaparaju Sivaraj

Software Engineer

Contact

Email: pinnaparajusivaraj@gmail.com

Phone: +91 7385098035

LinkedIn: Click here

Career Objective

I am an engineering student at Woxsen University. As a second-year B.Tech student majoring in Computer Science and Engineering, I am passionate about exploring the dynamic and ever-evolving field of technology. My academic coursework has provided me with a strong foundation in core computer science concepts, programming languages, and problem-solving skills.

Education

Education	Details	Grade
Current: Bachelor's Degree	Woxsen University - Computer Science Expected Graduation Year: 2026	1st year CGPA - 3.47/4
Secondary Schooling	Sri Chaitanya	10 CGPA
Intermediate	Sri Chaitanya	81%

Skills

Technical:

- HTML5
- CSS3
- Java
- Python
- SQL
- Mongo DB

Non-Technical:

- Quick Learner
- Adaptability
- Hardworking
- Leadership
- Presentation
- Communication
- Team-work

Languages

- Telugu
- Hindi
- English
- Currently learning Spanish

Projects

- Static Travel Tourism website named Info Jetsetter
- Theory: Method to reduce Exoskeleton price

7. **Reference:** https://www.coursera.org/learn/web-development

8. **Conclusion:** The HTML CV is well-structured, clear, and easy to navigate, with sections like Contact, Career Objective, Education, Skills, Languages, and Projects. The content is concise, and the use of an image adds a personal touch.

Experiment 2

1. **Aim:** Apply CSS on your CV webpage to make it more appealing
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

```
<!DOCTYPE html>
<html>
<head>
  <title>Sivaraj CV</title>
  <style>
    body {
      font-family: 'Arial', sans-serif;
    }
```

```
 #header {
   background-color: blueviolet;
   color: white;
   padding: 40px;
   font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
   text-align: center;
}

#header img {
   width: 135px;
   height: 140px;
   margin-right: 20px;
   border: 2px solid white;
   border-radius: 70px;
}

 #content {
    margin-left: auto;
    padding: 20px;
    display: flex;
 }

 #left-content {
    width: 25%;
    border-right: 3px solid blueviolet;
    padding-right: 10px;

 }
```

```
#right-content {
    padding-left: 20px;
}

h1, h2, h3 {
    color: blueviolet;
}

a {
    color: blue;
}

ul {
    list-style-type: square;
}

b {
    color: teal;
}

table {
    width: 80%;
    margin-top: 20px;
}

th, td {
    padding: 10px;
```

```
            text-align: left;
            border-bottom: 1px solid white;
        }

        th {
            background-color: blueviolet;
            color: white;
        }
    </style>
</head>

<body>
    <div id="header">
     <img src="https://media.licdn.com/dms/image/D4E03AQH7av2efH2AhA/profile-displayphoto-shrink_200_200/0/1673330337366?e=2147483647&v=beta&t=V84jZ9-1HxUhYiRXQtTA6RS_Ubwc0iOb4iNAv6NxSR8" alt="passport photo" align="left" />
     <div>
      <h1 style="color: white; text-align: center;">Pinnaparaju Sivaraj</h1>
      <p align="center" style="color: white;">Student At  Woxsen University</p>
     </div>
    </div>

    <hr>

    <div id="content">
        <div id="left-content">
            <h2>Contact</h2>
            <p>Email: pinnaparajusivaraj@gmail.com</p>
```

```
        <p>Phone: +91 7385098035</p>
        <p>LinkedIn:<a href="https://in.linkedin.com/in/pinnaparaju-sivaraj-06a489261"> Click here</a></p>

        <h2>Languages</h2>
        <ul>
            <li>Telugu</li>
            <li>Hindi</li>
            <li>English</li>
            <li>Currently learning Spanish</li>
        </ul>
    </div>

    <div id="right-content">
        <h2>Career Objective</h2>
        <p align="left"> I am an engineering student at Woxsen University. As a B.Tech student majoring in Computer Science and Engineering, I am passionate about exploring the dynamic and ever-evolving field of technology. My academic coursework has provided me with a strong foundation in core computer science concepts, programming languages, and problem-solving skills. </p>

        <h2>Education</h2>
        <table border="1">
            <tr>
                <th>Education</th>
                <th>Details</th>
                <th>Grade</th>
            </tr>
            <tr>
```

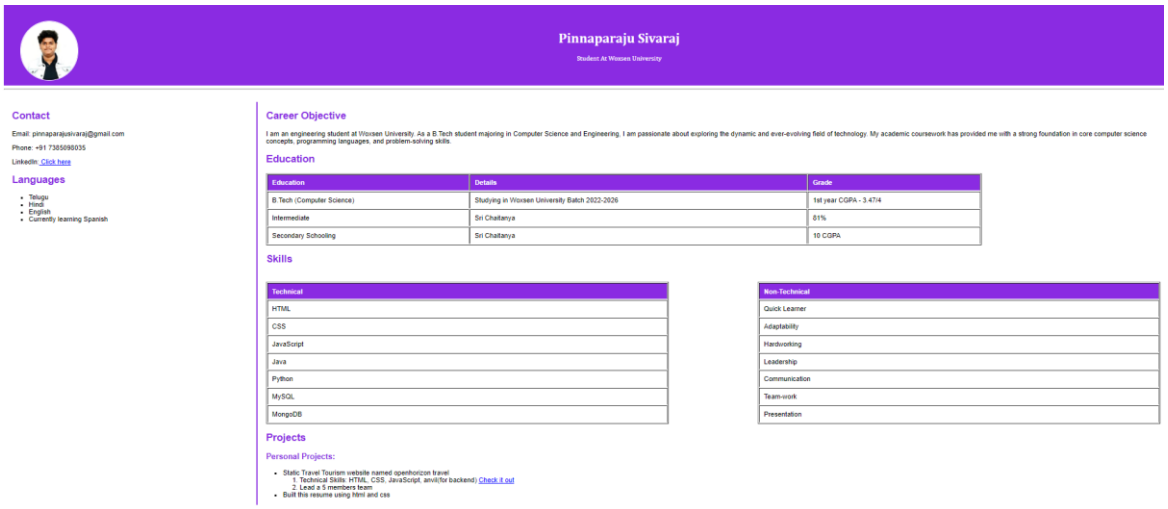
```
            <td>B.Tech (Computer Science)</td>
            <td>Studying in Woxsen University Batch 2022-2026</td>
            <td>1st year CGPA - 3.47/4</td>
        </tr>
        <tr>
            <td>Intermediate</td>
            <td>Sri Chaitanya</td>
            <td>81%</td>
        </tr>
        <tr>
            <td>Secondary Schooling</td>
            <td>Sri Chaitanya</td>
            <td>10 CGPA</td>
        </tr>

    </table>

    <h2>Skills</h2>
    <div style="display: flex; justify-content: space-between;">
        <table border="1" style="width: 45%;">
        <tr>
            <th>Technical</th>
        </tr>
        <tr>
            <td>HTML</td>
        </tr>
        <tr>
            <td>CSS</td>
```

```
    </tr>
    <tr>
        <td>JavaScript</td>
    </tr>
    <tr>
        <td>Java</td>
    </tr>
    <tr>
        <td>Python</td>
    </tr>
    <tr>
        <td>MySQL</td>

    </tr>
    <tr>
        <td>MongoDB</td>
    </tr>
</table>

<table border="1" style="width: 45%;">
    <tr>
        <th>Non-Technical</th>
    </tr>
    <tr>
        <td>Quick Learner</td>
    </tr>
    <tr>
        <td>Adaptability</td>
```

```
            </tr>
            <tr>
                <td>Hardworking</td>
            </tr>
            <tr>
                <td>Leadership</td>
            </tr>
            <tr>
                <td>Communication</td>
            </tr>

            <tr>
                <td>Team-work</td>

            </tr>
            <tr>
                <td>Presentation</td>
            </tr>
        </table>
    </div>

        <h2>Projects</h2>
        <h3>Personal Projects:</h3>
        <ul>
            <li>Static Travel Tourism website named openhorizon travel
            <ol>
                <li>Technical Skills: HTML, CSS, JavaScript, anvil(for backend) <a
href="https://openhorizontravel.netlify.app/">Check it out</a></li>
```

```
                <li>Lead a 5 members team</li>
            </ol>
            </li>
            <li>Built this resume using html and css</li>
        </ul>
    </div>
</div>
</body>
</html>
```

5. **Observations:** Clean and structured with a header, contact info, education, skills, and projects sections, Blue violet theme with clear division using flexbox Enhance mobile responsiveness, improve image alt text, and refine table styling
6. **Result:**



7. **Reference:** https://www.coursera.org/learn/web-development
8. **Conclusion:** The CV is well-structured, visually appealing, and organized logically. The use of color and layout contributes to a professional look, while the content is clear and concise. A few minor adjustments, such as improving responsiveness, enhancing the

image alt text, and tweaking the table styling, can make this CV even more polished and user-friendly

Experiment 3

1. **Aim:** Create Home Page with two sub-pages one for login and one for registration, validate is using JS
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

```
<!DOCTYPE html>
<head>
  <title>Login Page</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f4f4f4;
```

```
}

header {
    background-color: #333;
    padding: 10px;
    color: white;
    text-align: center;
}

nav {
    background-color: #ddd;
    padding: 10px;
    text-align: center;
}

nav a {
    text-decoration: none;
    color: #333;
    margin: 0 15px;
}

nav a:hover {
    color: #555;
}

form {
    max-width: 400px;
    margin: 20px auto;
```

```
    padding: 20px;
    background-color: #fff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

label {
    display: block;
    margin-bottom: 8px;
}

input {
    width: 100%;
    padding: 8px;
    margin-bottom: 15px;
    box-sizing: border-box;
}

button {
    background-color: #333;
    color: white;
    padding: 10px 15px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}

button:hover {
```

```
            background-color: #555;
        }

        .error {
            color: red;
            font-size: 14px;
        }

        .signup-link {
            margin-top: 10px;
            text-align: center;
        }

        .signup-link a {
            color: #333;
        }

        .signup-link a:hover {
            color: #555;
        }
    </style>
</head>
<body>
    <header>
        <h1>Login Page</h1>
    </header>

    <nav>
```

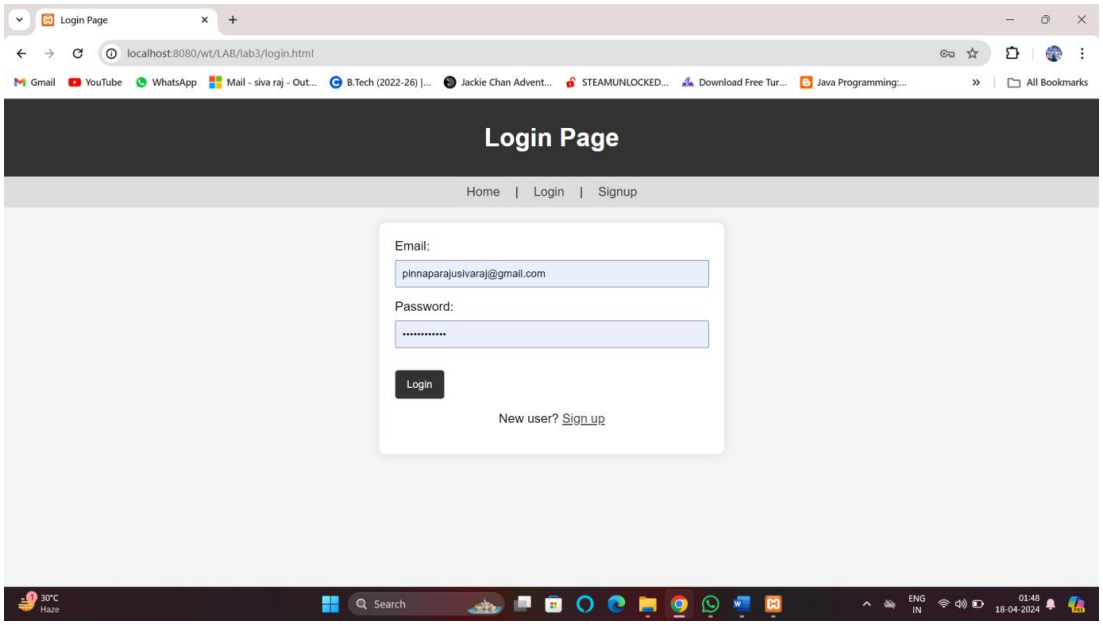
```
        <a href="index.html">Home</a> |
        <a href="#">Login</a> |
        <a href="sign-up.html">Signup</a>
    </nav>

    <form id="loginForm">
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" onchange="validateEmail()"
required>

        <label for="password">Password:</label>
        <input type="password" id="password" name="password" pattern="^(?=.*[a-
z])(?=.*[A-Z])(?=.*[@$!%*?&])[A-Za-z@$!%*?&]{6,}$" title="Password must
contain at least one uppercase letter, one lowercase letter, and one special character,
and be at least 6 characters long" required>

        <p class="error" id="emailError"></p>
        <p class="error" id="passwordError"></p>

        <button type="button" onclick="validateForm()">Login</button>

        <div class="signup-link">
            <p>New user? <a href="sign-up.html">Sign up</a></p>
        </div>
    </form>

    <script>
        function validateForm() {
            var email = document.getElementById("email").value;
```

```
        var password = document.getElementById("password").value;

        var emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;
        var passwordPattern = /^(?=.*[a-z])(?=.*[A-Z])(?=.*[@$!%*?&])[A-Za-z0-9@$!%*?&]{6,}$/;

        var emailError = document.getElementById("emailError");
        var passwordError = document.getElementById("passwordError");

        // Reset error messages
        emailError.textContent = "";
        passwordError.textContent = "";

        if (!email || !password) {
            alert("All fields are required. Please fill in all the fields.");
            return false; // Prevent form submission
        }

        if (!emailPattern.test(email)) {
            emailError.textContent = "Please enter a valid email address.";
            return false; // Prevent form submission
        }

        if (!passwordPattern.test(password)) {
            passwordError.textContent = "Password must contain at least one uppercase letter, one lowercase letter, and one special character, and be at least 6 characters long.";
            return false; // Prevent form submission
        }
```

```
            // If all validations pass, allow form submission
            alert("Login successful! Redirecting to home page...press ok");
            window.location.href = "index.html";
            return true;
        }
    </script>

</body>
</html>
```

5. **Observations:** Simple, clean login page with a header, navigation bar, and form Validates email and password with real-time error messages and pattern checks The form uses alert for success, which could be improved with a smooth redirection or a message confirmation

6. **Result:**



7. **Reference:** Introduction to Web Development | Coursera

8. **Conclusion:** The login page is well-structured with clear functionality, including email and password validation. The design is simple and user-friendly, but could be enhanced with smoother interactions, such as displaying success messages rather than relying on alert(). Overall, it provides a solid foundation for a login page.

Experiment 4

1. **Aim:** Write an HTML page including any required JavaScript that takes a number from one text field in the range of 0 to 999 and shows it in another text field in words. if the number is out of range, it should show “out of range” and if it is not a number, it should show “not a number” message in the result box
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

```
<!DOCTYPE html>
<html lang="en">
<head>
 <title>Number to Words Converter</title>
 <style>
  body {
   font-family: Arial, sans-serif;
   display: flex;
   justify-content: center;
```

```
    align-items: center;
    height: 100vh;
    margin: 0;
    background-image: url("background.jpg"); /* Replace with your desired background image URL */
    background-size: cover;
    background-position: center;
  }

  .container {
    background-color: rgba(255, 255, 255, 0.8);
    padding: 100px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
    animation: enter 0.5s ease-in-out forwards;
  }


  h2 {
    text-align: center;
    margin-bottom: 20px;
  }

  label {
    display: block;
    margin-bottom: 5px;
  }
```

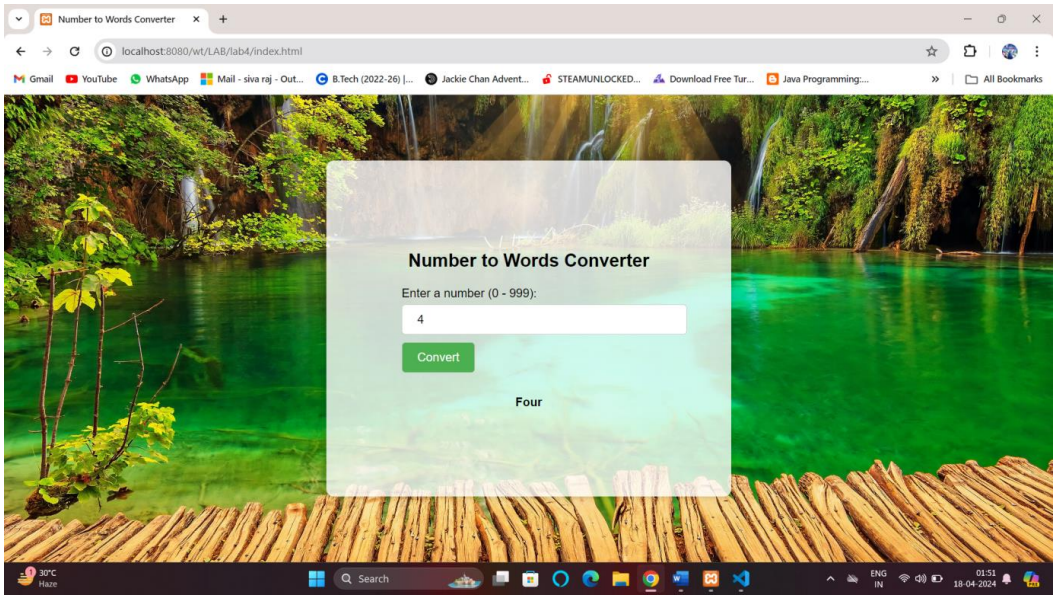
```
#number,
button {
  padding: 10px 20px;
  border: 1px solid #ccc;
  border-radius: 5px;
  margin-bottom: 10px;
  font-size: 16px;
  cursor: pointer;
}

#number {
  width: 100%;
}

button {
  background-color: #4CAF50;
  color: white;
  transition: background-color 0.2s ease-in-out;
}

button:hover {
  background-color: #3e8e41;
}

#output {
  margin-top: 20px;
  font-weight: bold;
  text-align: center;
```

```
  }

  .error {
    color: red;
    font-weight: bold;
    animation: shake 0.5s ease-in-out;
  }

 </style>
</head>
<body>
 <div class="container">
  <h2>Number to Words Converter</h2>
  <label for="number">Enter a number (0 - 999):</label>
  <input type="text" id="number" name="number" maxlength="3">
  <button onclick="convertToWords()">Convert</button>
  <div id="output"></div>
 </div>

 <script>
  function convertToWords() {
   var numberInput = document.getElementById("number").value;
   var outputDiv = document.getElementById("output");

   // Validate input
   if (!/^\d{1,3}$/.test(numberInput)) {
    outputDiv.innerHTML = '<p class="error">Please enter a valid number (0 - 999).</p>';
    return;
```

```
    }

    var number = parseInt(numberInput, 10);

    if (number < 0 || number > 999) {
      outputDiv.innerHTML = '<p class="error">Number must be in the range of 0 to 999.</p>';
      return;
    }

    // Array of words for numbers 0 to 19
    var units = ["Zero", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Ten",
      "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"];
    // Array of words for multiples of 10
    var tens = ["", "", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"];

    // Array of words for hundreds
    var hundreds = ["", "One Hundred", "Two Hundred", "Three Hundred", "Four Hundred", "Five Hundred", "Six Hundred", "Seven Hundred", "Eight Hundred", "Nine Hundred"];

    var words = "";

    if (number === 0) {
      words = units[number];
    } else {
      if (number >= 100) {
```

```
    words += hundreds[Math.floor(number / 100)] + " ";
    number %= 100;
   }

   if (number >= 20) {
    words += tens[Math.floor(number / 10)] + " ";
    number %= 10;
   }

   if (number > 0) {
    words += units[number];
   }
  }

  outputDiv.innerHTML = '<p>' + words + '</p>';
 }
</script>

</body>
</html>
```

5. **Observations:** The code is designed to convert numbers (from 0 to 999) into their corresponding English words. It takes user input, validates it, and then displays the result in words. The input is validated to ensure that the entered value is a number between 0 and 999. If the input is invalid (not a number or out of the range), an error message is shown.
6. **Result:**



7. **Reference:** **Programming Foundations with JavaScript, HTML and CSS | Coursera**
8. **Conclusion:** The Number to Words Converter functions as expected, with clean input validation and an intuitive user interface. The code efficiently handles numbers between 0 and 999, converting them into words with proper checks for input errors. The use of animations and clear styling makes the page visually appealing. It could be enhanced by adding support for larger numbers or improving error handling for edge cases (e.g., empty input).

Experiment 5

1. **Aim:** Create a PHP page to connect to DB (MySQL) and fetch records from an existing table and display on the php page.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

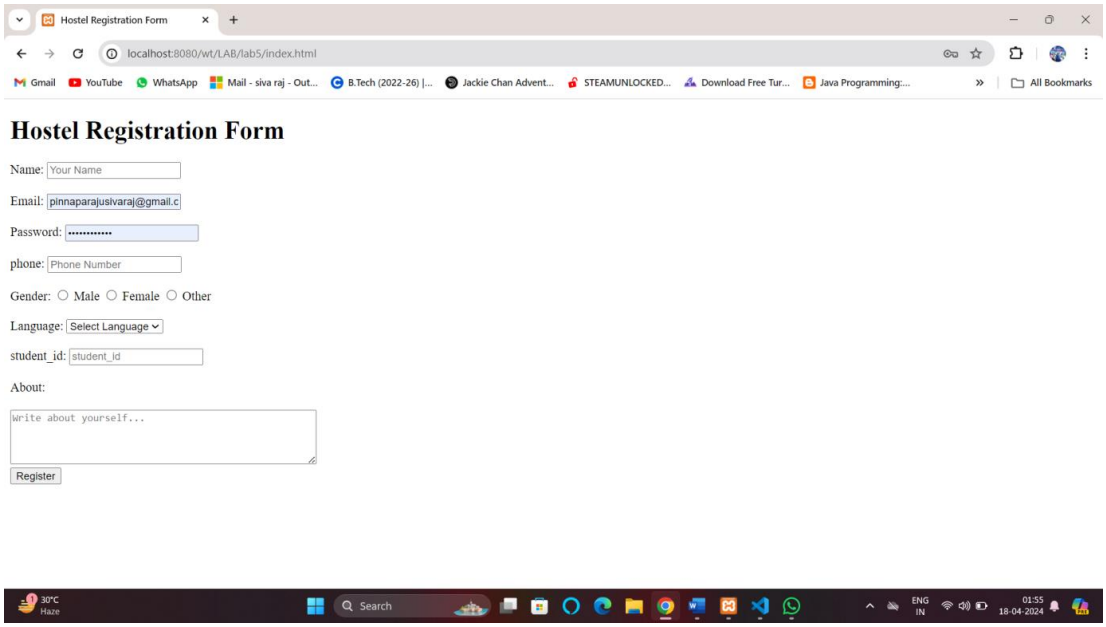
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Hostel Registration Form</title>
</head>
<body>
    <h1>Hostel Registration Form</h1>
```

```
<form action = "insert_student.php" method="post">
    <div>
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" placeholder="Your Name"><br><br>
    </div>
    <div>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" placeholder="Your Email"><br><br>
    </div>
    <div>
        <label for="password">Password:</label>
        <input type="password" id="password" name="password" placeholder="Password"><br><br>
    </div>
    <div>
        <label for="phone">phone:</label>
        <input type="tel" id="phone" name="phone" placeholder="Phone Number"><br><br>
    </div>
    <div>
        <label for="gender">Gender:</label>
        <input type="radio" id="gender" name="gender" value="male">
        <label for="male">Male</label>
        <input type="radio" id="gender" name="gender" value="female">
        <label for="female">Female</label>
        <input type="radio" id="gender" name="gender" value="other">
        <label for="other">Other</label><br><br>
```

```
        </div>
        <div>
            <label for="language">Language:</label>
            <select id="language" name="language">
                <option value="">Select Language</option>
                <option value="english">English</option>
                <option value="spanish">Spanish</option>
                <option value="french">French</option>
                <option value="german">German</option>
            </select><br><br>
        </div>
        <div>
            <label for="student_id">student_id:</label>
            <input type="text" id="student_id" name="student_id"
placeholder="student_id"><br><br>
        </div>
        <div>
            <label for="about">About:</label><br><br>
            <textarea id="about" name="about" rows="4" cols="50"
placeholder="Write about yourself..."></textarea>
        </div>
        <div>
            <button type="submit">Register</button>
        </div>


    </form>
</body>
```

```
</html>
```

5. **Observations:** The HTML form captures essential details like Name, Email, Password, Phone, Gender, Language, Student ID, and a textarea for additional information (About). The form data is sent via POST method to insert_student.php, which is likely the server-side script handling the submission
6. **Result:**

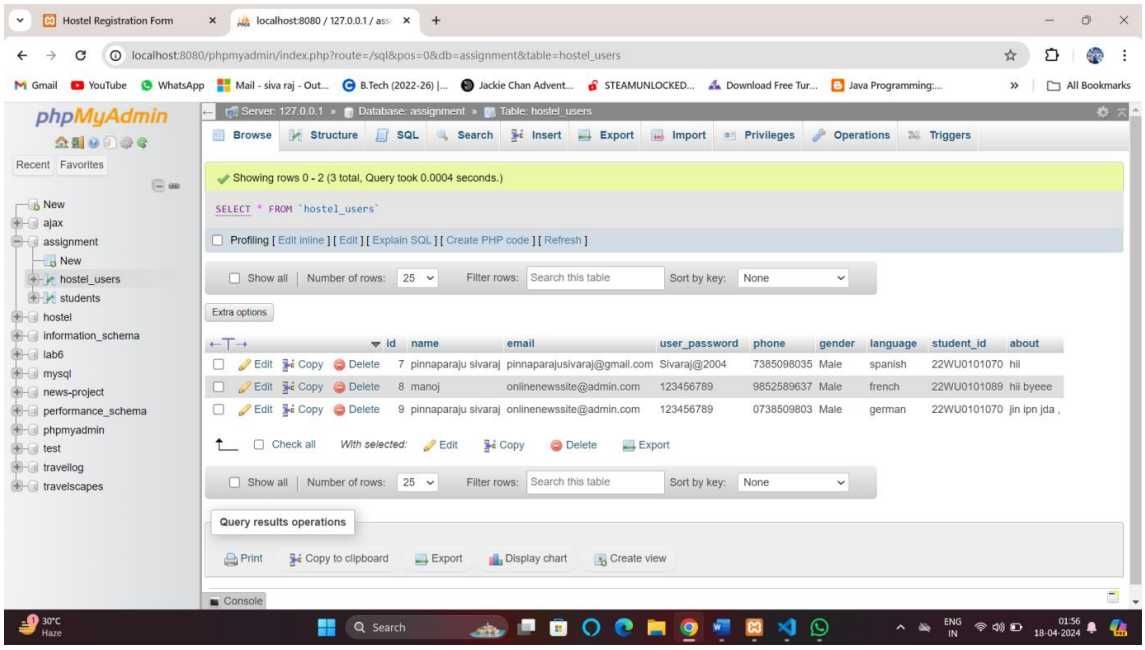


Insert_student.php:

```
<?php

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $name = $_POST['name'];
    $email = $_POST['email'];
    $user_password = $_POST['password'];
    $phone = $_POST['phone'];
```

```
    $gender = $_POST['gender'];
    $language = $_POST['language'];
    $student_id = $_POST['student_id'];
    $about = $_POST['about'];

    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "assignment";

    $conn = new mysqli($servername, $username, $password, $dbname);

    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    $sql = "INSERT INTO hostel_users (name, email, user_password, phone,
gender, language, student_id, about) VALUES ('$name', '$email',
'$user_password', '$phone', '$gender', '$language', '$student_id', '$about')";

    if ($conn->query($sql) === TRUE) {
        echo "Registered Successfully";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

    $conn->close();
}
```

?>



7. **Reference:** [Programming Foundations with JavaScript, HTML and CSS | Coursera]

8. **Conclusion:** The code provides a basic hostel registration form with fields for personal details and preferences. However, there is a minor issue with duplicate id attributes for the radio buttons, which should be corrected to ensure proper functionality and validation. The form is well-structured and should work efficiently for capturing user data

Experiment 6

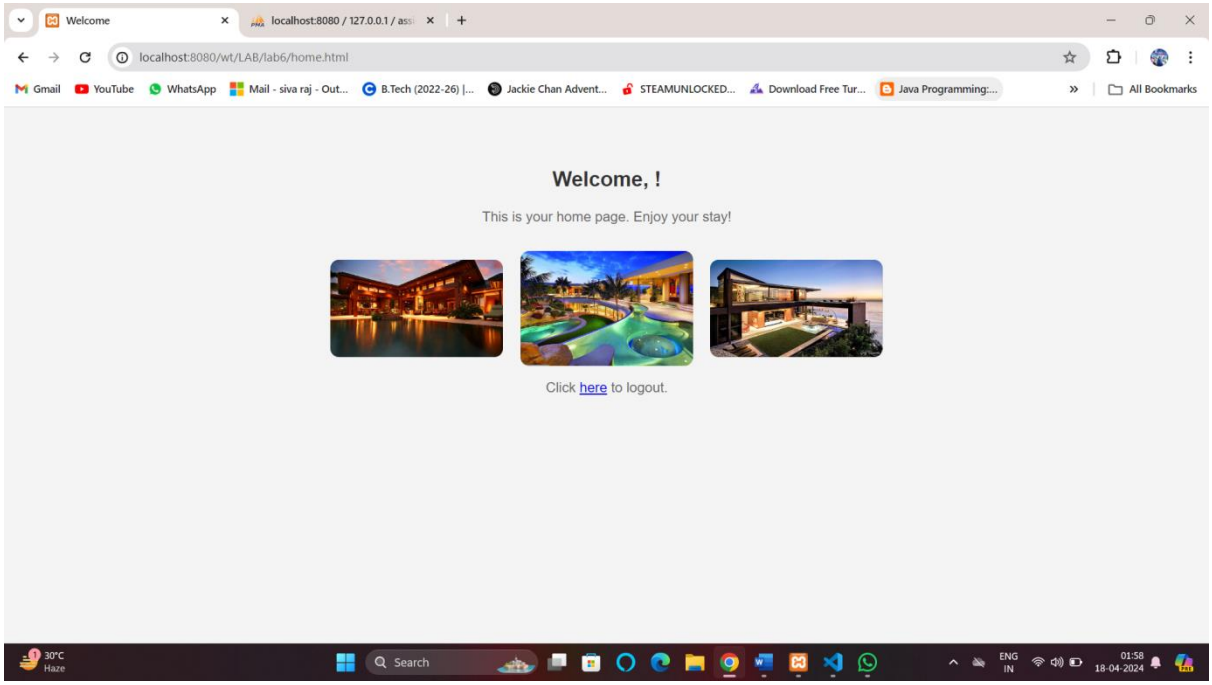
1. **Aim:** Using PHP and the previous login and registration page, and on successful registration store all the information in DB and on valid login show welcome page and on invalid login details show error message.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

Home.html:

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f4f4f4;
    }
    .container {
      width: 80%;
      margin: auto;
```

```
            padding-top: 50px;
            text-align: center;
        }
        h2 {
            color: #333;
        }
        p {
            color: #666;
        }
        .image-container {
            margin-top: 30px;
            display: flex;
            justify-content: center;
            align-items: center;
        }
        .image-container img {
            width: 200px;
            margin: 0 10px;
            border-radius: 10px;
        }
    </style>
</head>
<body>

<div class="container">
    <h2>Welcome, <?php echo "Welcome, " . $username . "!"; ?>!</h2>
    <p>This is your home page. Enjoy your stay!</p>
    <div class="image-container">
```

```
        <img src="home1.jpg" alt="Placeholder Image 1">
        <img src="home2.jpg" alt="Placeholder Image 2">
        <img src="home3.jpg" alt="Placeholder Image 3">
    </div>
    <p>Click <a href="logout.php">here</a> to logout.</p>
</div>


</body>
</html>
```



Login.html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Login Page</title>
```

```
<style>
    body {
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        background-color: #f4f4f4;
    }
    .container {
        width: 80%;
        margin: auto;
        padding-top: 50px;
    }
    h2 {
        text-align: center;
        color: #333;
    }
    form {
        width: 50%;
        margin: auto;
        background: #fff;
        padding: 20px;
        border-radius: 5px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    input[type="text"],
    input[type="password"],
    input[type="submit"] {
        width: 100%;
```

```
            padding: 10px;
            margin-bottom: 10px;
            border: 1px solid #ccc;
            border-radius: 5px;
            box-sizing: border-box;
        }
        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            cursor: pointer;
        }
        input[type="submit"]:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>

<div class="container">
    <h2>Login</h2>
    <form action="login.php" method="post">
        Username: <input type="text" name="username"><br>
        Password: <input type="password" name="password"><br>
        <input type="submit" value="Login">
    </form>
    <p style="text-align: center;">Not registered yet? <a
href="register.html">Register</a></p>
</div>
```

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

Login.php:

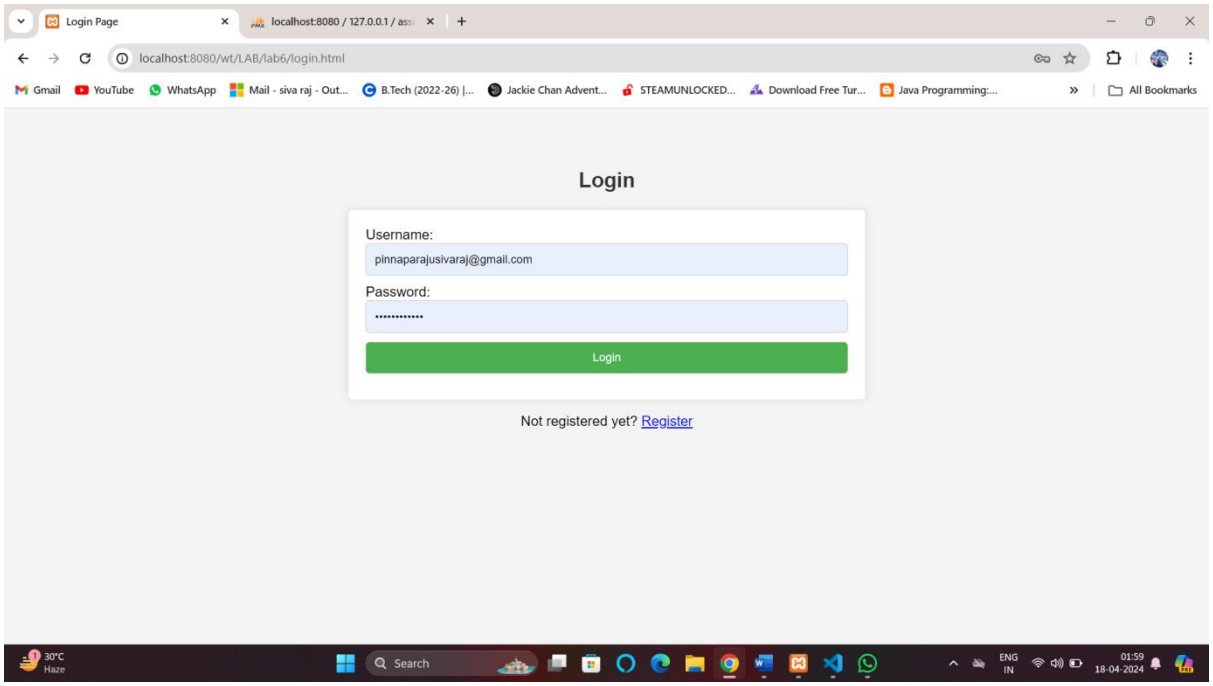
```
<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab6";

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];

    // Retrieve user data from database
    $sql = "SELECT * FROM users WHERE username='$username'";
    $result = $conn->query($sql);
```

```
    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        if (password_verify($password, $row['password'])) {
            echo "Welcome, " . $username . "!";
            // Redirect to home page
            header("Location: home.html");
            exit();
        } else {
            echo "Invalid login details.";
        }
    } else {
        echo "Invalid login details.";
    }
}

$conn->close();
?>
```

Logout.php:

```
<?php
// Start the session
session_start();

// Unset all of the session variables
$_SESSION = array();

// Destroy the session
session_destroy();
```

```
// Redirect to the login page after logout
header("Location: login.html");
exit;
?>
```

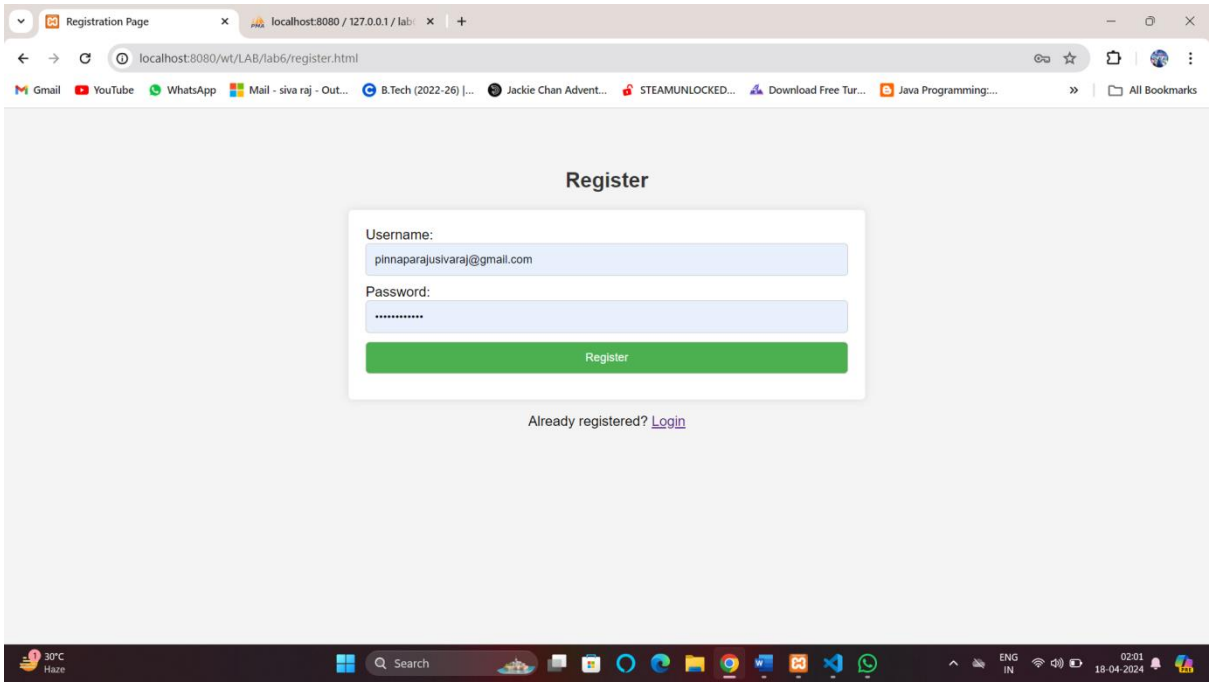


Register.html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Registration Page</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
```

```
    padding: 0;
    background-color: #f4f4f4;
}
.container {
    width: 80%;
    margin: auto;
    padding-top: 50px;
}
h2 {
    text-align: center;
    color: #333;
}
form {
    width: 50%;
    margin: auto;
    background: #fff;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
input[type="text"],
input[type="password"],
input[type="submit"] {
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
```

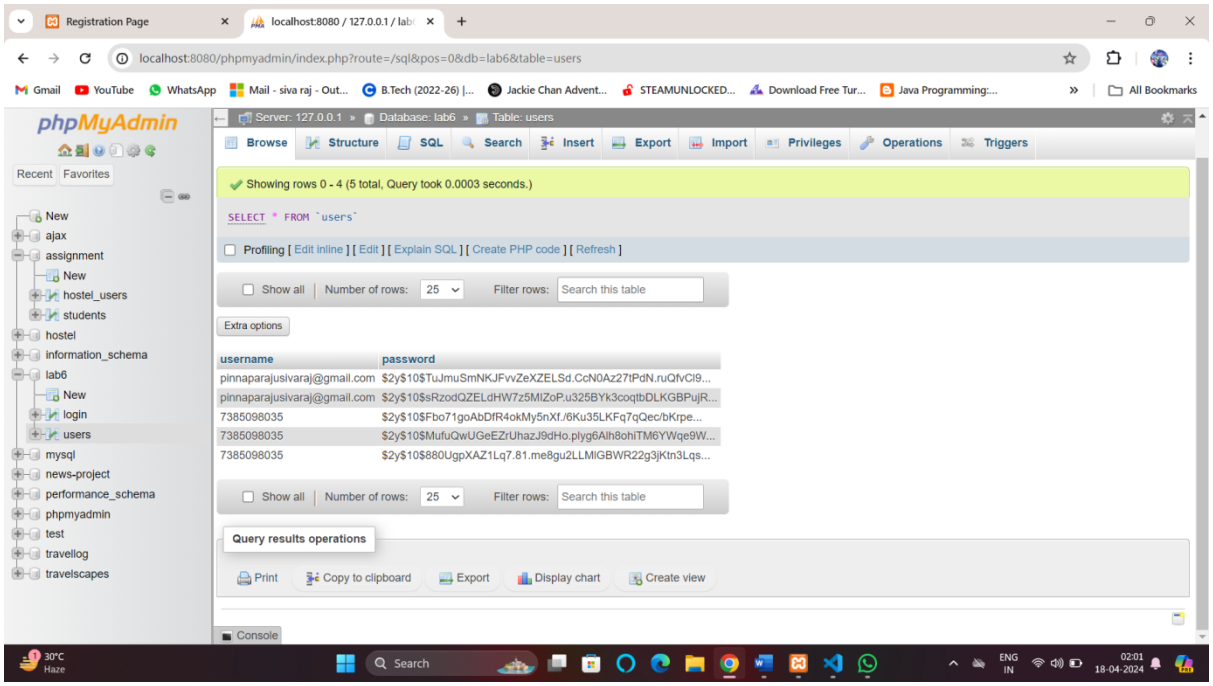
```
            box-sizing: border-box;
        }
        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            cursor: pointer;
        }
        input[type="submit"]:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>

<div class="container">
    <h2>Register</h2>
    <form action="register.php" method="post">
        Username: <input type="text" name="username"><br>
        Password: <input type="password" name="password"><br>
        <input type="submit" value="Register">
    </form>
    <p style="text-align: center;">Already registered? <a
href="login.html">Login</a></p>
</div>

</body>
</html>
```



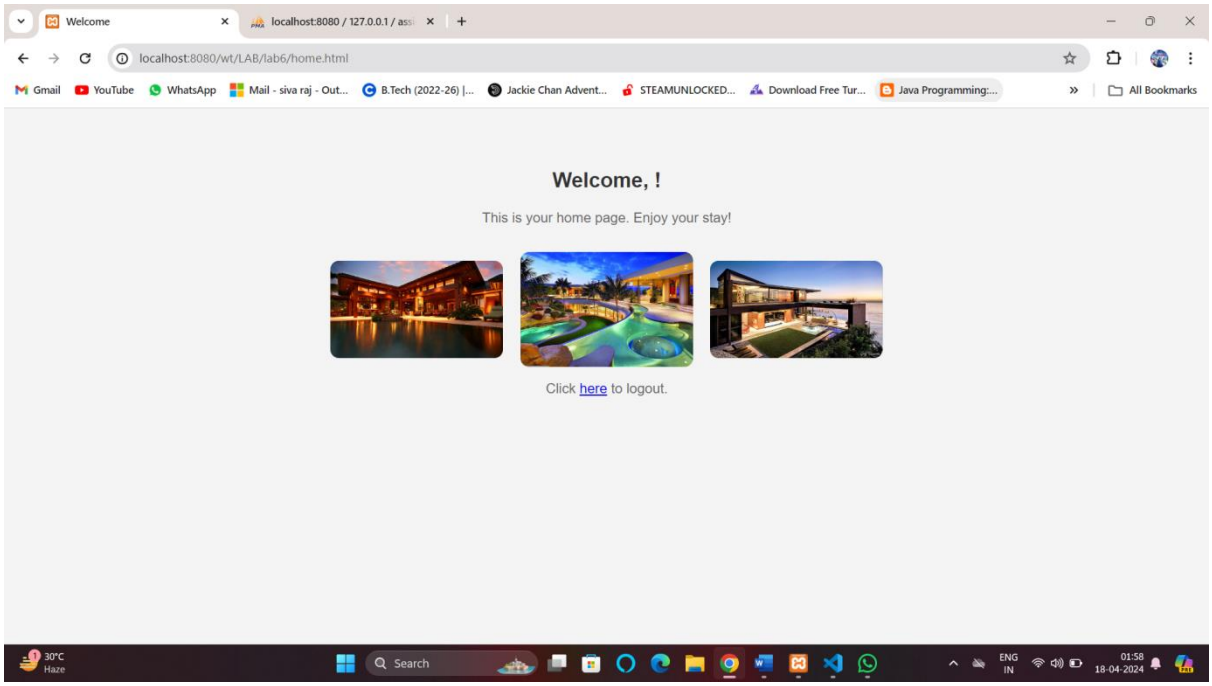
Register. php:

```
<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab6";

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```
// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = password_hash($_POST['password'], PASSWORD_DEFAULT); // Hash the password

    // Insert user data into database
    $sql = "INSERT INTO users (username, password) VALUES ('$username', '$password')";

    if ($conn->query($sql) === TRUE) {
        echo "Registration successful!";
        header("Location: login.html");
        exit();
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
}

$conn->close();
?>
```



5. **Observations:** The registration page is designed to collect various user details such as name, email, password, phone number, gender, language, student ID, and a personal description. This data is submitted to a PHP script (insert_student.php) via a POST request. The login page validates the credentials (email and password) using PHP. If the login details are correct, the user is redirected to a welcome page; otherwise, an error message is displayed prompting the user to enter valid credentials. Additionally, a MySQL database is used to store the user information securely, with password hashing to ensure security. Error handling is implemented to display messages for invalid login attempts, guiding users to correct their input.
6. **Result:**



7. **Reference:** Programming Foundations with JavaScript, HTML and CSS | Coursera

8. **Conclusion:** To successfully implement the registration and login process, it is essential to securely store user details in a database. Input validation should occur both on the client side (using HTML forms) and the server side (using PHP) to ensure data integrity and security. Passwords must be hashed before storing them in the database, ensuring that sensitive information remains protected. In case of incorrect login attempts, clear error messages should be provided to help users resolve any issues. This approach ensures a smooth and secure user experience

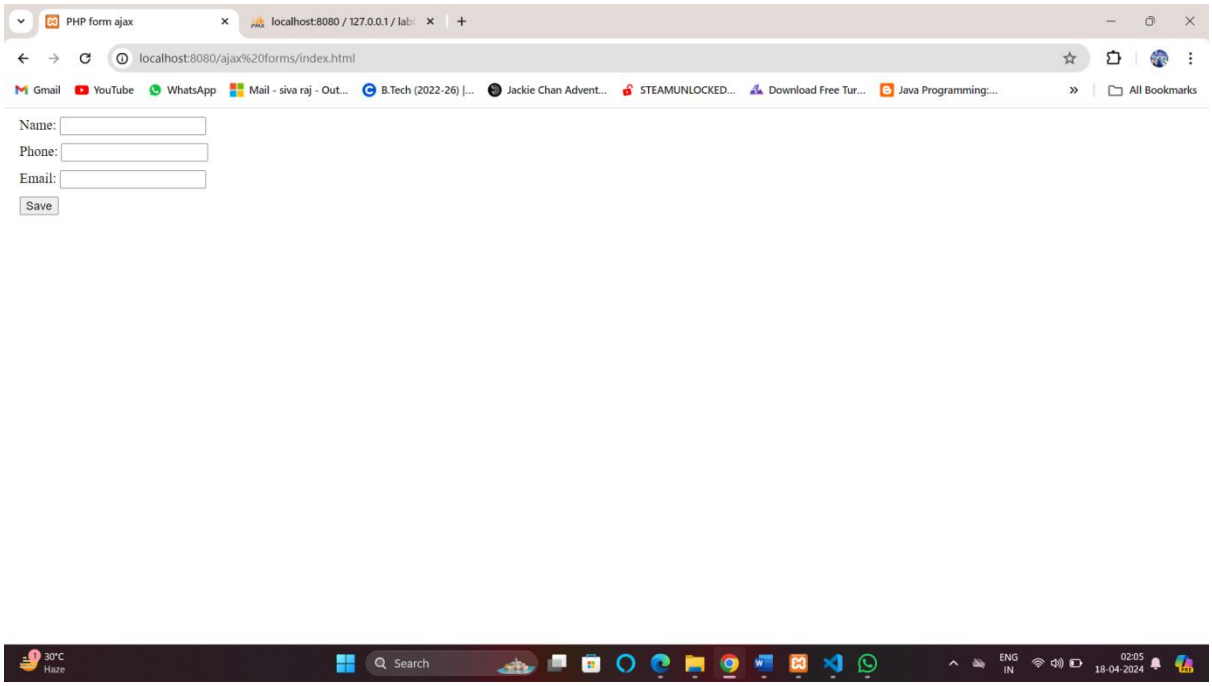
Experiment 7

1. **Aim:** Create a JSP page to connect to DB (MySQL) and fetch records from an existing table and display on the php page.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

Index.html:

```
<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab6";

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
```

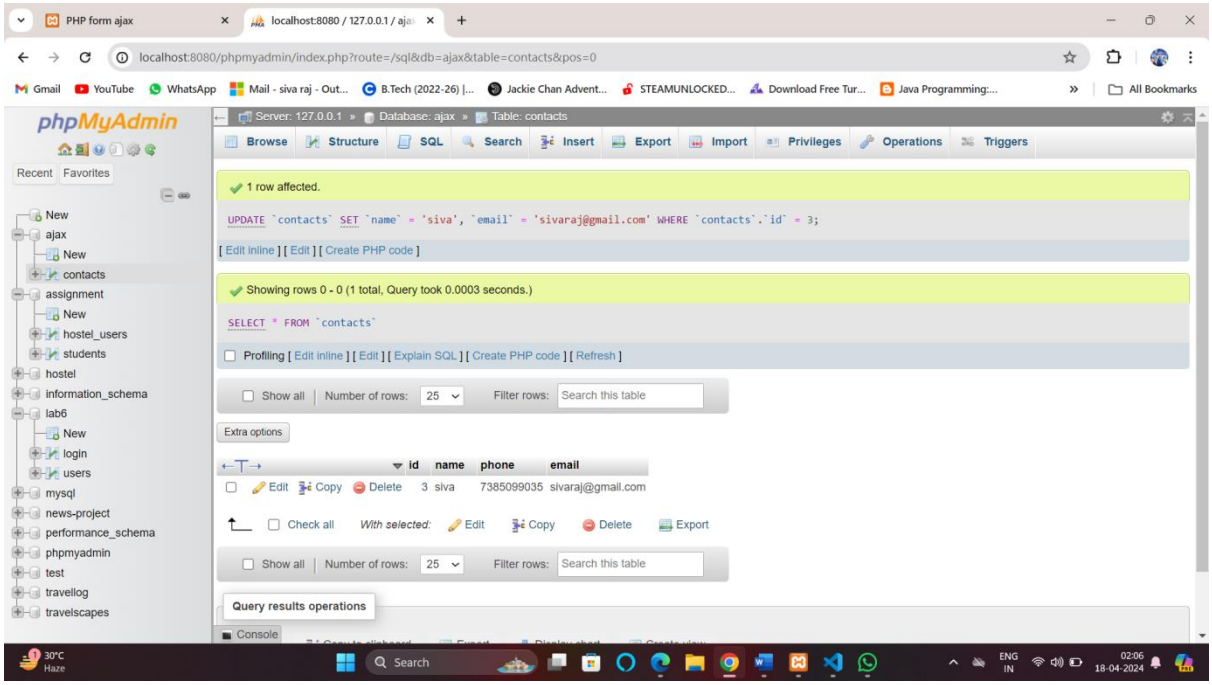
```
    $password = password_hash($_POST['password'],
PASSWORD_DEFAULT); // Hash the password


    // Insert user data into database
    $sql = "INSERT INTO users (username, password) VALUES ('$username',
'$password')";


    if ($conn->query($sql) === TRUE) {
        echo "Registration successful!";
        header("Location: login.html");
        exit();
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
}

$conn->close();
?>
```



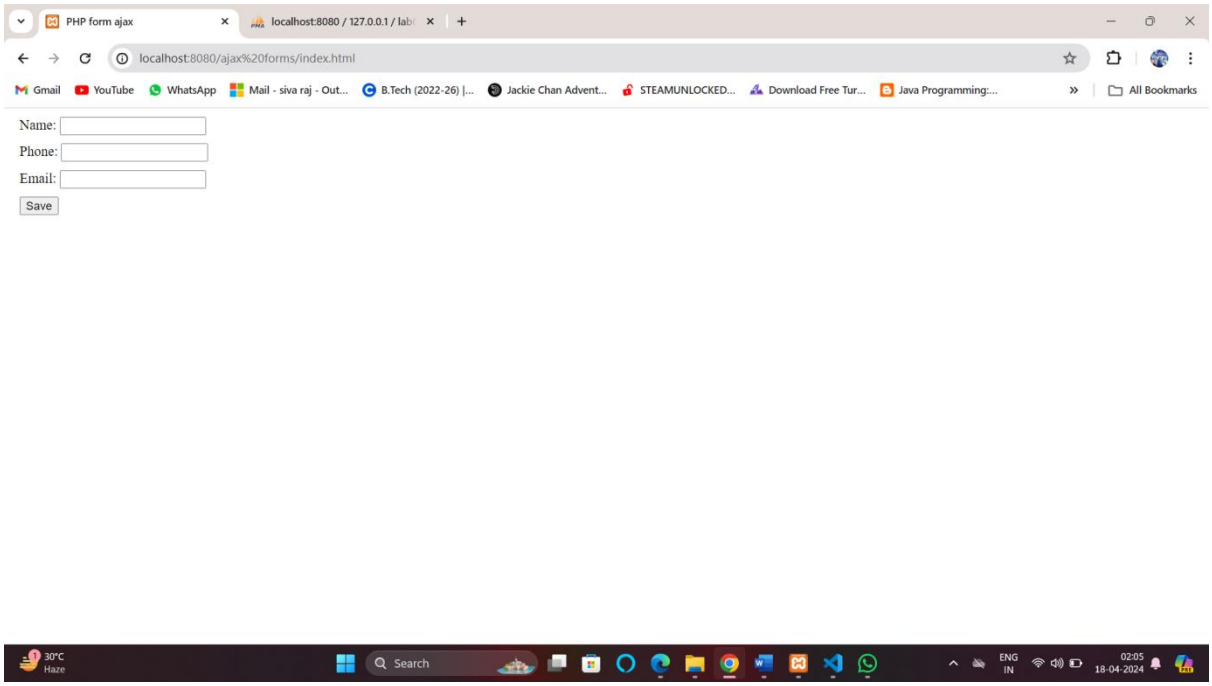
Test.php:

```
<?php
// Database connection
$servername = "localhost";
$username = "root"; // Change to your database username
$password = ""; // Change to your database password
$dbname = "ajax"; // Change to your database name

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
```

```
}

// SQL query to insert data
$sql = "INSERT INTO contacts (name, phone, email) VALUES ('varenya',
'7385099035', 'varenyavarma@gmail.com')";
if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
// Close connection
$conn->close();
?>
```



5. **Observations:** In the scenario of creating a JSP page to connect to a MySQL database and display records, the process involves several key steps. Firstly, the JSP page establishes a connection to the MySQL database using JDBC (Java Database Connectivity). The connection is made by specifying the database URL, username, and password, along with loading the MySQL JDBC driver class. Once the connection is established, an SQL query (e.g., SELECT * FROM table_name) is executed to fetch the data from the database. The result is stored in a ResultSet object, which is then iterated to display the records within an HTML structure, usually in the form of a table. For instance, if the query retrieves user data, the columns such as id, name, and email can be displayed as rows in the table. Proper error handling is essential throughout this process to catch any exceptions that may occur during the connection, query execution, or data fetching process.
6. **Result:**



7. **Reference:** **Programming Foundations with JavaScript, HTML and CSS | Coursera**
8. **Conclusion:** The JSP page successfully integrates with the MySQL database using JDBC to retrieve and display records dynamically on the web page. The key steps of

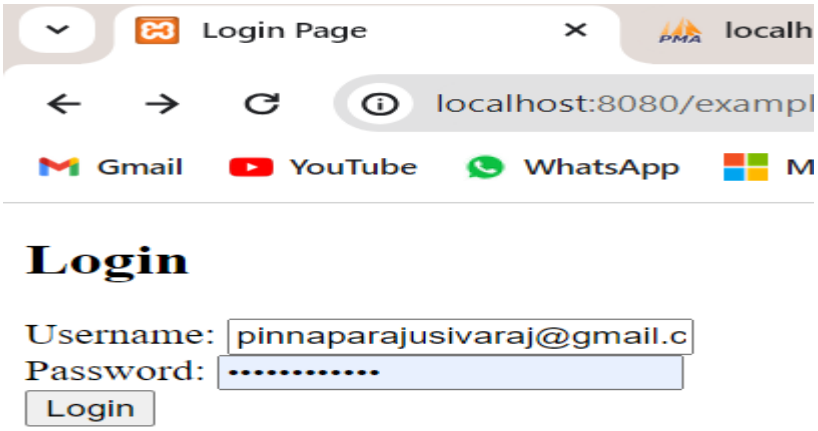
database connection, query execution, and result display are effectively handled within the JSP file. The use of the ResultSet to fetch data and output it as HTML content ensures that the records are displayed in a user-friendly manner. Additionally, incorporating error handling allows for better debugging and user feedback in case of connection or query issues. This approach provides a seamless way to display database records in a web application, showcasing how JSP can be used to interact with databases to serve dynamic content to users.

Experiment 8

1. **Aim:** Using JSP and the previous login and registration page, and on successful registration store all the information in DB and on valid login show welcome page and on invalid login details show error message.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

Login.jsp:

```
<!DOCTYPE html>
<html>
<head>
    <title>Login Page</title>
</head>
<body>
    <h2>Login</h2>
    <form action="loginProcess.jsp" method="post">
        Username: <input type="text" name="username" required><br>
        Password: <input type="password" name="password" required><br>
        <input type="submit" value="Login">
    </form>
</body>
</html>
```



Loginprocess.jsp:

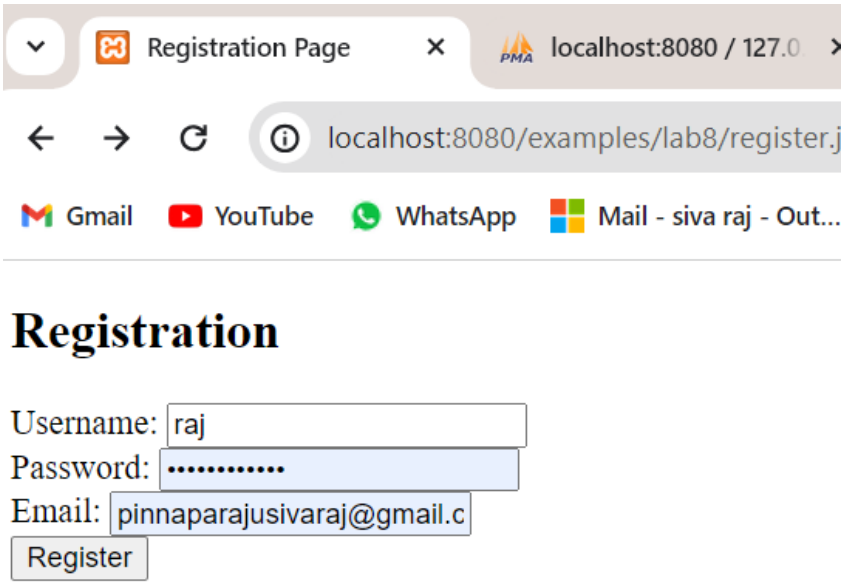
```
<%@ page import="java.sql.*" %>
<%@ page import="javax.sql.*" %>
<%@ page import="javax.naming.*" %>
<%@ page import="java.io.*" %>

<%
    // Retrieve form data
    String username = request.getParameter("username");
    String password = request.getParameter("password");

    // Establish database connection
    try {
        Context ctx = new InitialContext();
```

```
        DataSource ds = (DataSource)
ctx.lookup("java:comp/env/jdbc/your_database"); // Change 'your_database' to
your datasource name

        Connection conn = ds.getConnection();


        // Prepare SQL statement

        PreparedStatement ps = conn.prepareStatement("SELECT * FROM users
WHERE username=? AND password=?");

        ps.setString(1, username);

        ps.setString(2, password);


        // Execute SQL statement

        ResultSet rs = ps.executeQuery();


        if (rs.next()) {

            // Valid login, redirect to welcome page

            response.sendRedirect("welcome.jsp");

        } else {

            // Invalid login, show error message

            out.println("Invalid username or password. <a href='login.jsp'>Try
again</a>");

        }


        // Close connections

        rs.close();

        ps.close();

        conn.close();

    } catch (Exception e) {

        out.println("Error: " + e.getMessage());
```

```
    }
%>
```

Register.jsp:

```
<!DOCTYPE html>
<html>
<head>
    <title>Registration Page</title>
</head>
<body>
    <h2>Registration</h2>
    <form action="registerProcess.jsp" method="post">
        Username: <input type="text" name="username" required><br>
        Password: <input type="password" name="password" required><br>
        Email: <input type="email" name="email" required><br>
        <input type="submit" value="Register">
    </form>
</body>
</html>
```



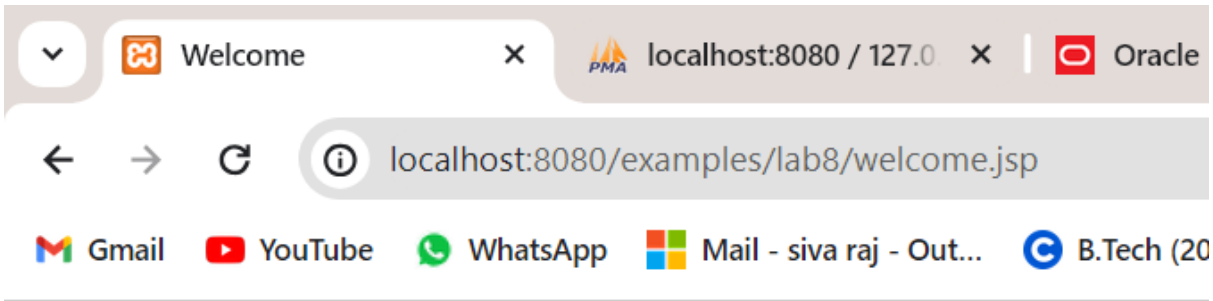
Register Process. jsp:

```
<%@ page import="java.sql.*" %>
<%@ page import="javax.sql.*" %>
<%@ page import="javax.naming.*" %>
<%@ page import="java.io.*" %>

<%
    // Retrieve form data
    String username = request.getParameter("username");
    String password = request.getParameter("password");
    String email = request.getParameter("email");

    // Establish database connection
    try {
        Context ctx = new InitialContext();
```

```
        DataSource ds = (DataSource) ctx.lookup("java:comp/env/jdbc/ajax"); // Change 'your_database' to your datasource name
        Connection conn = ds.getConnection();

        // Prepare SQL statement
        PreparedStatement ps = conn.prepareStatement("INSERT INTO users (username, password, email) VALUES (?, ?, ?)");
        ps.setString(1, username);
        ps.setString(2, password);
        ps.setString(3, email);

        // Execute SQL statement
        int rowsAffected = ps.executeUpdate();
        if (rowsAffected > 0) {
            out.println("Registration successful. <a href='login.jsp'>Login</a>");
        } else {
            out.println("Registration failed. Please try again.");
        }

        // Close connections
        ps.close();
        conn.close();
    } catch (Exception e) {
        out.println("Error: " + e.getMessage());
    }
%>
```

Welcome.jsp:

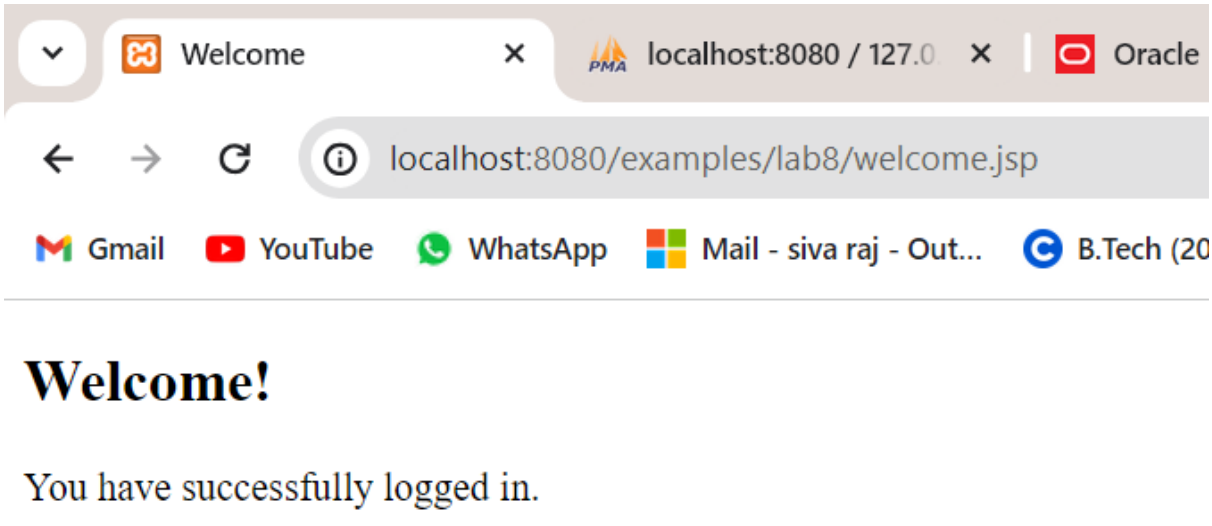
```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome</title>
</head>
<body>
  <h2>Welcome!</h2>
  <p>You have successfully logged in.</p>
</body>
</html>
```



Welcome!

You have successfully logged in.

5. **Observations:** In this JSP-based system, user registration and login functionalities are integrated with a MySQL database. Upon successful registration, user information is stored in the database, and login credentials are validated against the stored data. If the login is successful, the user is directed to a welcome page. Otherwise, an error message is shown to the user. This system uses JDBC for database interactions and JSP for dynamically displaying content.
6. **Result:**



7. **Reference:** Programming Foundations with JavaScript, HTML and CSS | Coursera

8. **Conclusion:** The JSP-based login and registration system effectively connects to the MySQL database for storing and retrieving user data. It provides a smooth user experience by handling valid logins with a welcome page and displaying error messages for invalid attempts. Proper session management and validation enhance security and reliability, ensuring a secure login process.

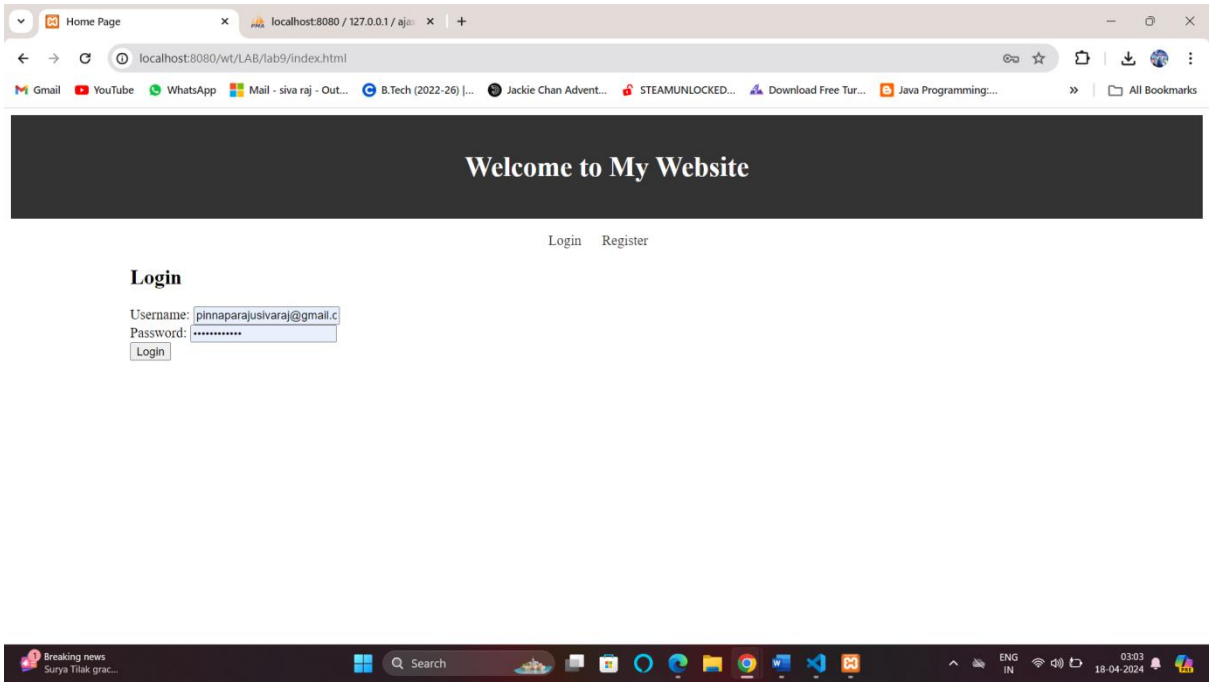
Experiment 9

1. **Aim:** Create Home Page with two sub-pages one for login and one for registration, validate it using Jquery.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

Index.html:

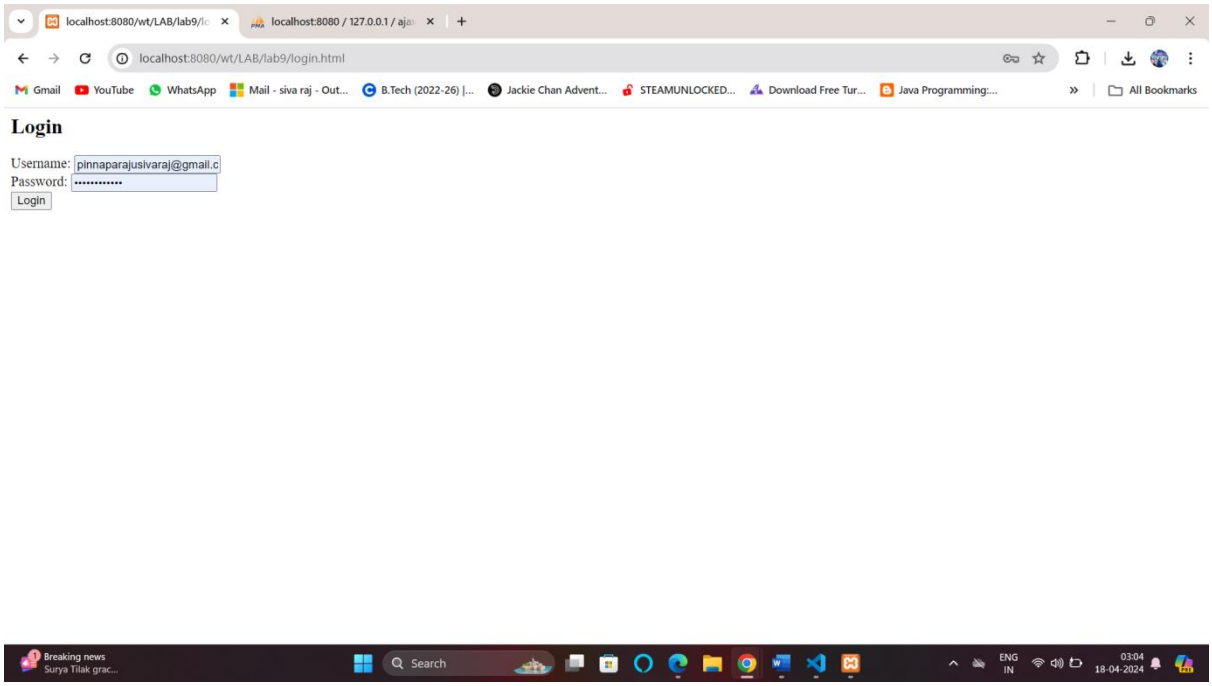
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home Page</title>
```

```
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script src="script.js"></script>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
    </header>
    <nav>
        <ul>
            <li><a href="login.html">Login</a></li>
            <li><a href="register.html">Register</a></li>
        </ul>
    </nav>
    <div id="content">
        <!-- Content will be loaded dynamically using jQuery -->
    </div>
</body>
</html>
```



Login.html:

```
<div id="login">
  <h2>Login</h2>
  <form id="loginForm">
    Username: <input type="text" id="username"><br>
    Password: <input type="password" id="password"><br>
    <input type="submit" value="Login">
  </form>
  <div id="loginError" class="error"></div>
</div>
```



Register.html:

```
<div id="register">
<h2>Register</h2>
<form id="registerForm">
    Username: <input type="text" id="regUsername"><br>
    Password: <input type="password" id="regPassword"><br>
    Email: <input type="email" id="email"><br>
    <input type="submit" value="Register">
</form>
<div id="registerError" class="error"></div>
</div>
```

Script.js:

```
$(document).ready(function() {
    // Load login page by default
    $('#content').load('login.html');
```

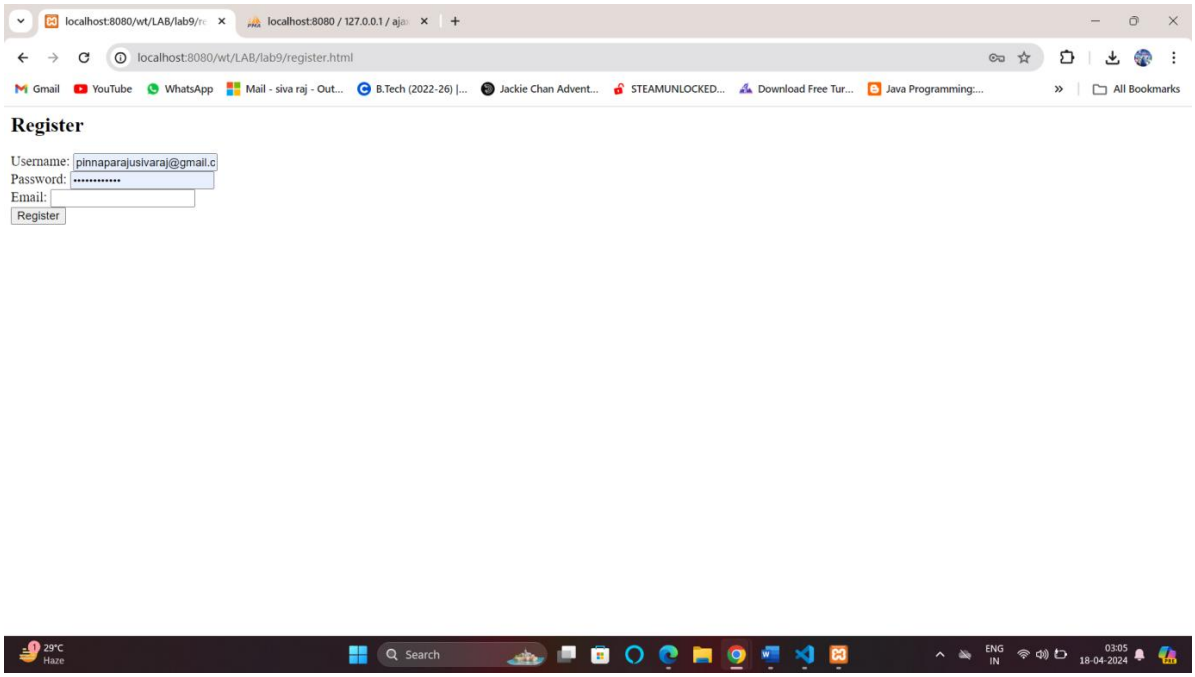
```
// Handle navigation clicks
$('nav a').click(function(e) {
    e.preventDefault();
    var page = $(this).attr('href');
    $('#content').load(page + '.html');
});

// Handle login form submission
$(document).on('submit', '#loginForm', function(e) {
    e.preventDefault();
    var username = $('#username').val();
    var password = $('#password').val();
    // Perform validation (e.g., check if fields are empty)
    if (!username || !password) {
        $('#loginError').text('Please fill in all fields.');
    } else {
        // Perform login process
        // For demonstration purposes, just display a success message
        $('#loginError').text('Login successful.');
    }
});

// Handle registration form submission
$(document).on('submit', '#registerForm', function(e) {
    e.preventDefault();
    var regUsername = $('#regUsername').val();
    var regPassword = $('#regPassword').val();
```

```
        var email = $('#email').val();
        // Perform validation (e.g., check if fields are empty)
        if (!regUsername || !regPassword || !email) {
            $('#registerError').text('Please fill in all fields.');
        } else {
            // Perform registration process
            // For demonstration purposes, just display a success message
            $('#registerError').text('Registration successful.');
        }
    });
});
```

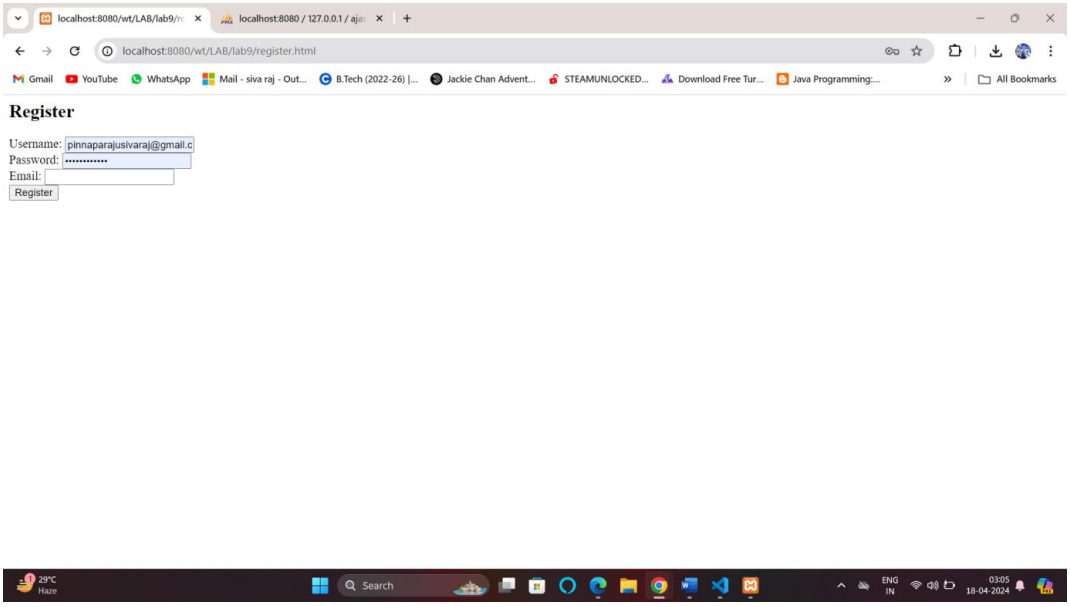
Styles.css:

```
/* Add your styles here */
header {
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 20px;
}

nav ul {
    list-style-type: none;
    padding: 0;
    text-align: center;
}

nav ul li {
```

```
    display: inline;
    margin-right: 20px;
}

nav ul li a {
    color: #333;
    text-decoration: none;
}

#content {
    width: 80%;
    margin: 20px auto;
}

.error {
    color: red;
}
```



5. **Observations:** The project involves creating a home page with two sub-pages, one for login and another for registration. jQuery is used for client-side validation of the login and registration forms. The validation checks include verifying correct email formats, password strength, and ensuring required fields are filled. This helps in preventing errors and incomplete data submission before the form reaches the server.
6. **Result:**



7. **Reference:** Programming Foundations with JavaScript, HTML and CSS | Coursera

8. **Conclusion:** By using jQuery for form validation, the user experience is enhanced, ensuring that only valid and properly formatted data is submitted. This approach reduces server-side errors and improves the efficiency and security of the login and registration processes.

Experiment 10

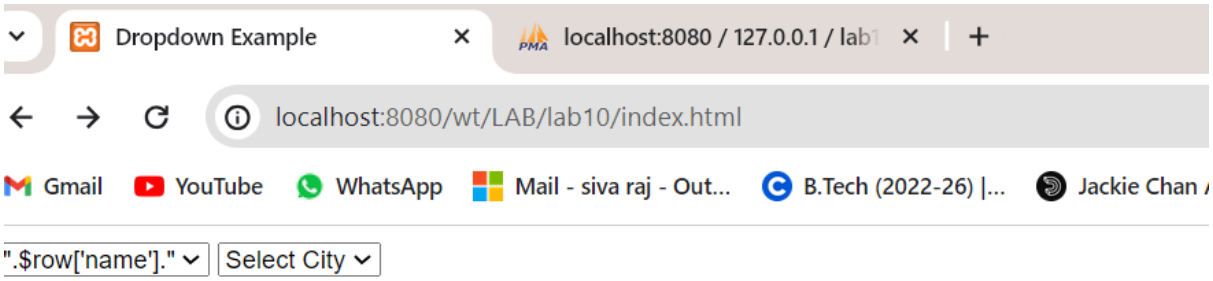
1. **Aim:** Create two dropdowns, the values of first dropdown are fetched from DB using PHP/JSP and onchnage of value load the content in another dropdown without refreshing/reloading the page (Hint: Use AJAX) - Ex: States List are fetched from DB and on change of State, load the city names of the repsective state in another dropdown.
2. **Theory:** This HTML code showcases a CV for Mahankali Siri Chandana Nagasai, a BTech student, featuring personal details, career objectives, education, technical skills (Python, Java, HTML, CSS, MongoDB, MySQL), and projects (AI traffic optimization, hydrogen fuel cells, internship). It is structured for a clean presentation of key information.
3. **Tools Required:**
   Text Editor (for coding):
   Examples: Visual Studio Code, Sublime Text, Notepad++
4. **Code:**

Index.html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Dropdown Example</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script
>
</head>
<body>


<select id="states">
    <option value="">Select State</option>
    <?php
    // Display states fetched from database
    while ($row = $states_result->fetch_assoc()) {
        echo "<option value='".$row['id']."'>".$row['name']."</option>";
```

```
    }
    ?>
</select>

<select id="cities">
    <option value="">Select City</option>
</select>

<script>
$(document).ready(function(){
    $('#states').change(function(){
        var state_id = $(this).val();
        if(state_id !== ''){
            // Send AJAX request to fetch cities
            $.ajax({
                url: 'get_cities.php',
                type: 'GET',
                data: {state_id: state_id},
                success: function(response){
                    $('#cities').html(response);
                }
            });
        } else {
            // Clear cities dropdown if no state is selected
            $('#cities').html('<option value="">Select City</option>');
        }
    });
});
```

```
</script>

</body>
</html>
```



Fetch cities. php:

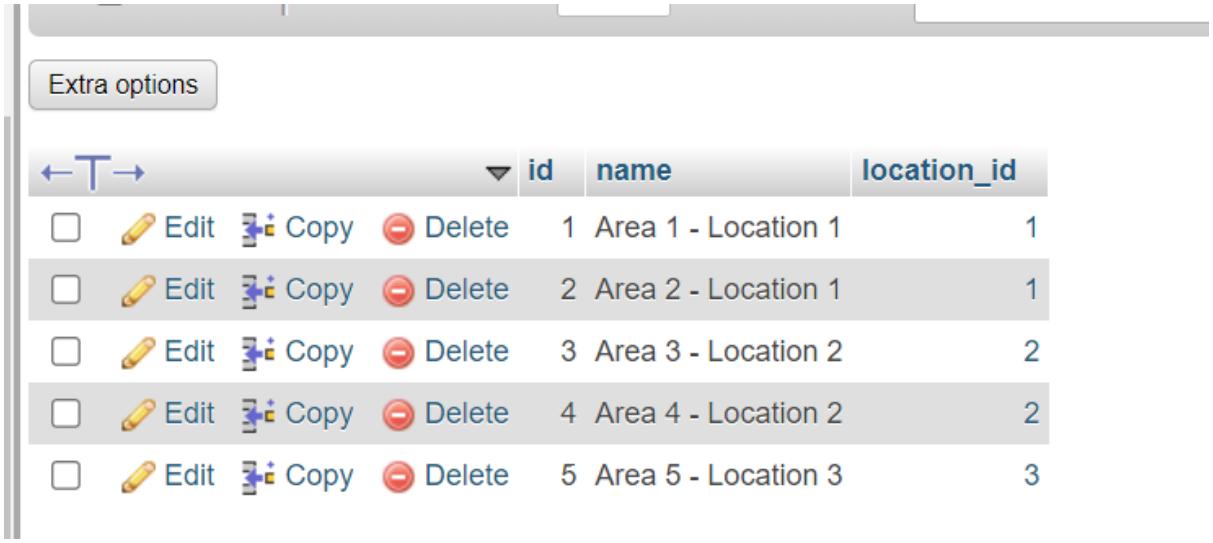
```
<?php

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab10t";

$conn = new mysqli($servername, $username, $password, $dbname);

if (isset($_GET['state_id'])) {
    $state_id = $_GET['state_id'];
    $cities_query = "SELECT * FROM cities WHERE state_id = $state_id";
    $cities_result = $conn->query($cities_query);

    // Generate HTML for cities dropdown
```

```
    $cities_dropdown = "<option value="">Select City</option>";
    while ($row = $cities_result->fetch_assoc()) {
        $cities_dropdown .= "<option
value="".$row['id].""">".$row['name']."</option>";
    }

    // Return HTML for cities dropdown
    echo $cities_dropdown;
}
?>
```

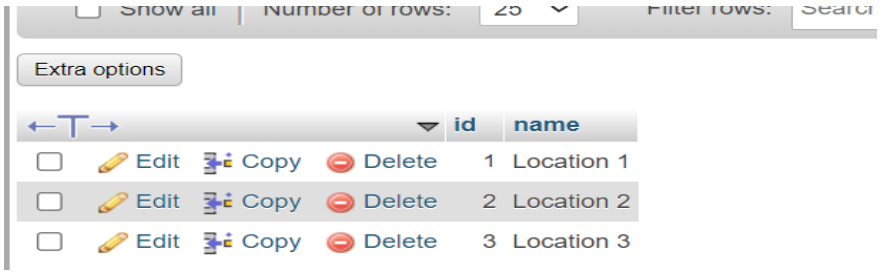
Fetch_states.php:

```
<?php
// Connect to your database
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab10t";

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Fetch states
```

```
$states_query = "SELECT * FROM states";
$states_result = $conn->query($states_query);

// Fetch cities based on selected state
if (isset($_GET['state_id'])) {
    $state_id = $_GET['state_id'];
    $cities_query = "SELECT * FROM cities WHERE state_id = $state_id";
    $cities_result = $conn->query($cities_query);
}
?>
```



5. **Observations:** The task involves creating two dropdowns where the first dropdown (such as a list of states) fetches data from a database using PHP or JSP. When the user selects a state, the second dropdown (e.g., cities) is dynamically populated with corresponding values through an AJAX request, without needing a page reload.
6. **Result:**



7. **Reference:** Programming Foundations with JavaScript, HTML and CSS | Coursera

8. **Conclusion:** Using AJAX for dynamically loading data into the second dropdown enhances the user experience by eliminating page refreshes, ensuring a smoother, more efficient interaction. It also optimizes server requests and reduces unnecessary network traffic.