LAB 1

Task 1: Create a simple page to demonstrate the usage of necessary html tags. The page should contains and their description (image and description). You can show your own creativity which will be evaluated accordingly.

Background:

HTML, which stands for Hyper Text Markup Language, is the backbone of the World Wide Web. It's a standardized markup language used for creating web pages and structuring the content within them. HTML documents consist of a series of elements or "tags" that define the structure and presentation of web content.

HTML tags are enclosed in angle brackets (< >) and are used to define various elements on a web page, such as headings, paragraphs, links, images, and more. These tags provide instructions to web browsers on how to render and display the content to users. Each HTML tag serves a specific purpose and can have attributes that modify its behavior or appearance.

For example, the <h1> tag is used to create a top-level heading, while the <p> tag is used for paragraphs of text.

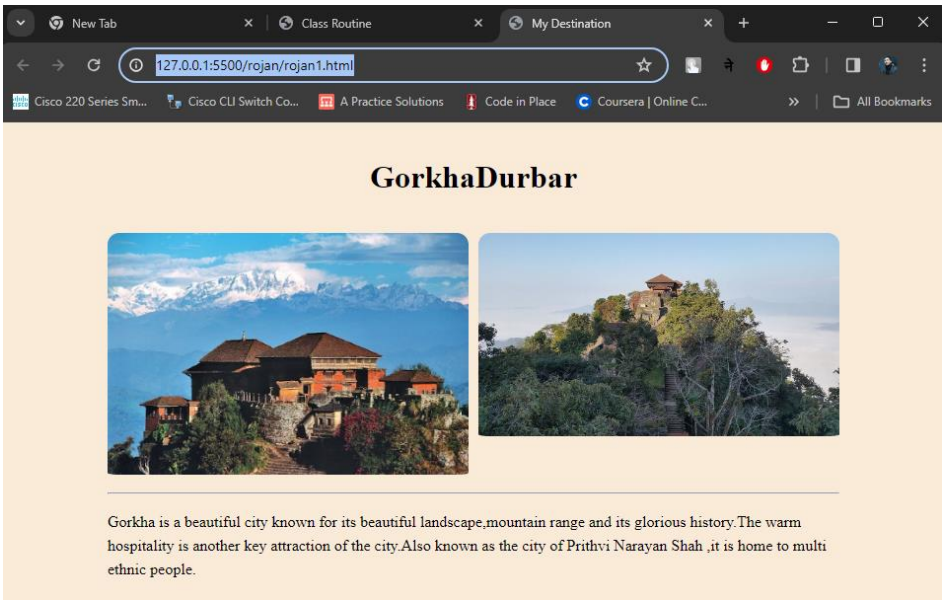
Source Code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>My Destination</title>
    <style>
      body {
        background-color: antiquewhite;
      }
      .destination-desc {
        padding: 0 100px;
      }
      .destination-desc h1 {
```

```
    text-align: center;
    font-weight: 800;
    padding: 16px 0;
  }
  .destination-img {
    display: grid;
    grid-template-columns: repeat(2, 1fr); /* Two columns of equal width */
    grid-gap: 10px;
  }
  .destination-desc img {
    width: 100%;
    max-height: 380px;
    object-fit: cover;
    border-radius: 12px;
    padding-bottom: 10px;
  }
  .destination-desc p {
    line-height: 150%;
  }
grid-template-columns: repeat(2, 1fr); /* Two columns of equal width */
    grid-gap: 10px;
  }
  .destination-desc img {
    width: 100%;
    max-height: 380px;
    object-fit: cover;
    border-radius: 12px;
    padding-bottom: 10px;
  }
  .destination-desc p {
    line-height: 150%;
```

```
    }
  </style>
 </head>
 <body>
  <div class="destination-desc">
   <h1>GorkhaDurbar</h1>
   <div class="destination-img">
    <img src="webtech.jpg" alt="" />
    <img src="webtech1.jpg" alt="" />
   </div>
   <hr />
   <p>
```
Gorkha is a beautiful city known for its beautiful landscape,mountain range and its glorious history.The warm hospitality is another key
attraction of the city.Also known as the city of Prithvi Narayan Shah ,it is home to multi ethnic people.
```
   </p>
  </div>
 </body>
</html>
```

Output


New Tab
Class Routine
My Destination
127.0.0.1:5500/rojan/rojan1.html
Cisco 220 Series Sm...
Cisco CLI Switch Co...
A Practice Solutions
Code in Place
Coursera | Online C...
All Bookmarks
GorkhaDurbar
Gorkha is a beautiful city known for its beautiful landscape,mountain range and its glorious history.The warm hospitality is another key attraction of the city.Also known as the city of Prithvi Narayan Shah ,it is home to multi ethnic people.

Task 2: Create a page that shows the course structure of B.Sc.CSIT using list. Also create organizational structure of College of Applied Business.

Background:

A list is a fundamental structural element used to organize and display related pieces of content in a structured manner. Lists are essential for improving the readability and organization of web documents. There are three list types in HTML:

- Unordered list - used to group a set of related items in no particular order
- Ordered list - used to group a set of related items in a specific order
- Description list - used to display name/value pairs such as terms and definitions

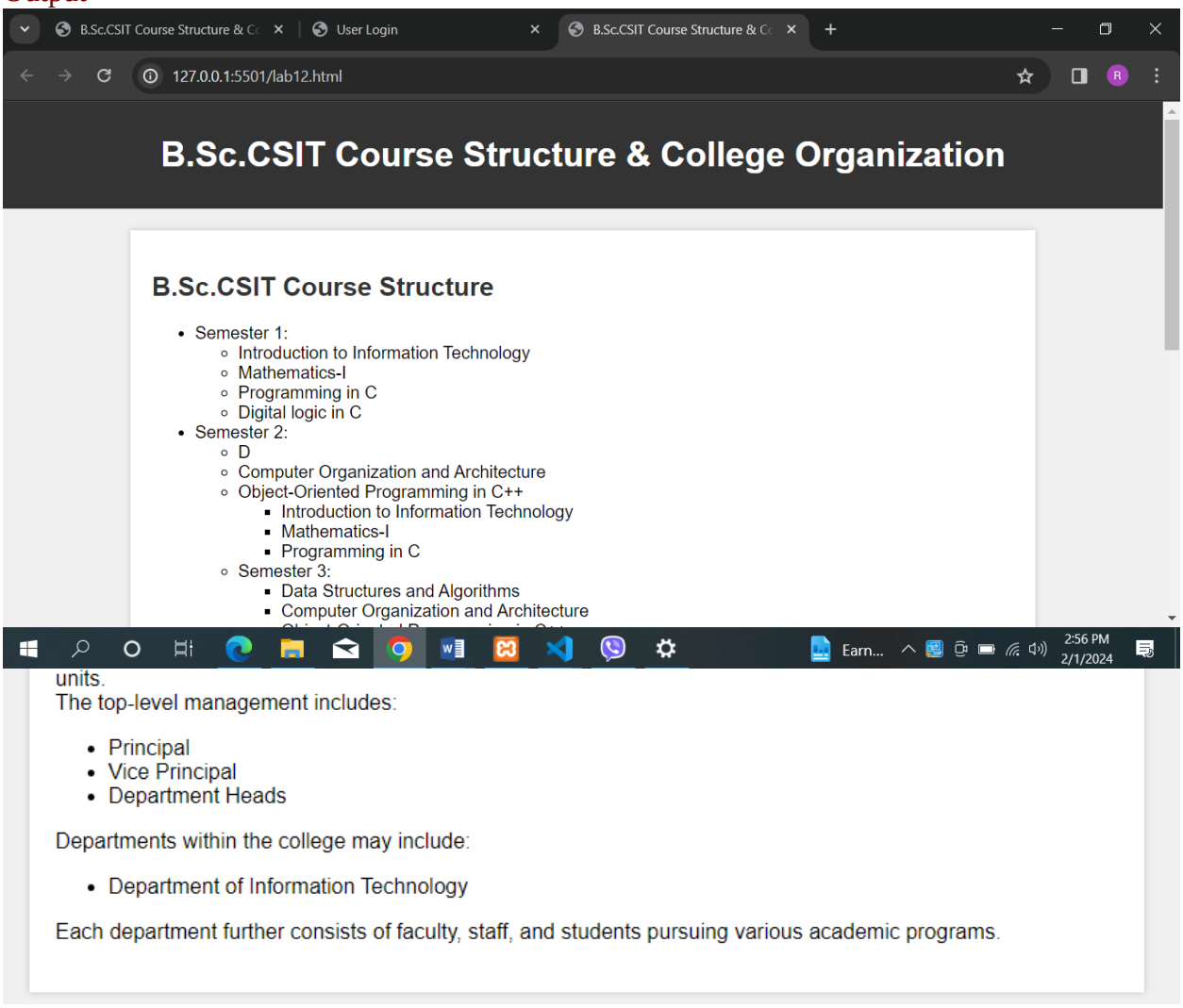
Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>B.Sc.CSIT Course Structure & College Organization</title>
  <style>
    /* Basic CSS for styling */
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f0f0f0;
    }
    header {
      background-color: #333;
      color: #fff;
      text-align: center;
      padding: 10px;
    }
    main {
      max-width: 800px;
      margin: 20px auto;
      padding: 20px;
      background-color: #fff;
      box-shadow: 0 0 5px rgba(0, 0, 0, 0.2);
    }
    h2 {
```

```
        color: #333;
      }
    </style>
</head>
<body>
    <header>
      <h1>B.Sc.CSIT Course Structure & College Organization</h1>
    </header>
    <main>
      <section>
        <h2>B.Sc.CSIT Course Structure</h2>
        <ul>
          <li>Semester 1:
            <ul>
              <li>Introduction to Information Technology</li>
              <li>Mathematics-I</li>
              <li>Programming in C</li>
              <li>Digital logic in C</li>
              <!-- Add more courses for Semester 1 -->
            </ul>
          </li>
          <li>Semester 2:
            <ul>
              <li>D</li>
              <li>Computer Organization and Architecture</li>
              <li>Object-Oriented Programming in C++</li>
            <ul>
              <li>Introduction to Information Technology</li>
              <li>Mathematics-I</li>
              <li>Programming in C</li>
              <!-- Add more courses for Semester 1 -->
            </ul>
          </li>
           <li>Semester 3:
            <ul>
              <li>Data Structures and Algorithms</li>
              <li>Computer Organization and Architecture</li>
              <li>Object-Oriented Programming in C++</li>
              <li>statistics in C++</li>
              <!-- Add more courses for Semester 2 -->
            </ul>
          </li>
      </section>
      <section>
        <h2>Organizational Structure of College of Applied Business</h2>
```

```
        <p>The College of Applied Business has a hierarchical organizational
                <li>Computer Organization and Architecture</li>
                <li>Object-Oriented Programming in C++</li>
                <!-- Add more courses for Semester 2 -->
              </ul>
            </li>
            <!-- Add more semesters and courses as needed -->
          </ul>
      </section>
      <section>
        <h2>Organizational Structure of College of Applied Business and Technology</h2>
           <p>The College of Applied Business has an organizational structure with various departments and administrative units.<br> The top-level management includes:</p>
        <ul>
          <li>Principal</li>
          <li>Vice Principal</li>
          <li>Department Heads</li>
        </ul>
        <p>Departments within the college may include:</p>
        <ul>
          <li>Department of Information Technology</li>
          <!-- Add more departments as needed -->
        </ul>
          <p>Each department further consists of faculty, staff, and students pursuing various academic programs.</p>
      </section>
  </main>
</body>
</html>
```

Output



units.
The top-level management includes:

- Principal
- Vice Principal
- Department Heads

Departments within the college may include:

- Department of Information Technology

Each department further consists of faculty, staff, and students pursuing various academic programs.

Task 3: Create a user registration form with necessary fields.

Background:

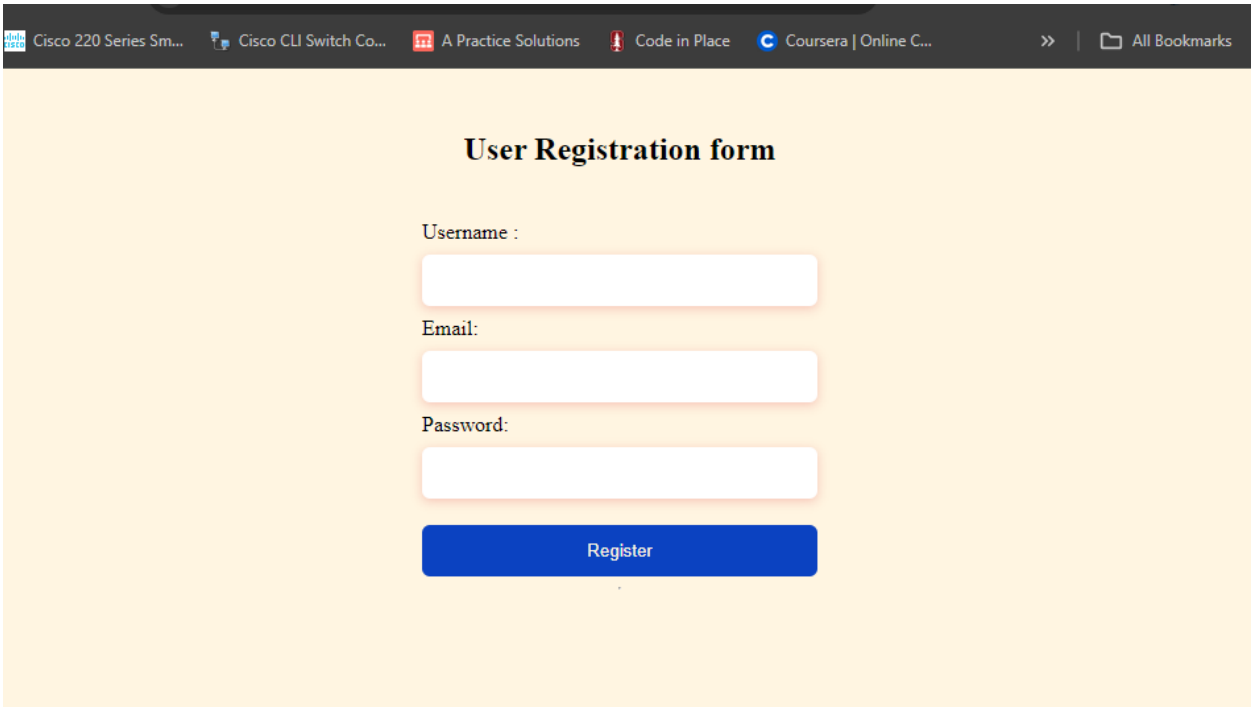
A HTML form is a crucial element used to collect and submit user input or data on a web page. It provides a structured way for website visitors to interact with the site by entering information, making selections, and submitting data for further processing, such as searching, logging in, or submitting contact information.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>User Login</title>
    <style>
      body {
        background-color: #caaa65;
      }
      .userlogin-form {
        padding: 24px 0;
        display: flex;
        flex-direction: column;
        align-items: center;
      }
      .userlogin-form h1 {
        font-size: 24px;
        font-weight: 800;
        padding-bottom: 24px;
      }
      .userlogin-form form {
        width: 300px;
      }
      .userlogin-form form .input-group {
        display: flex;
        flex-direction: column;
      }
      .userlogin-form form .input-group input {
        border-radius: 6px;
        padding: 12px 8px;
        margin: 8px 0;
        box-sizing: border-box;
```

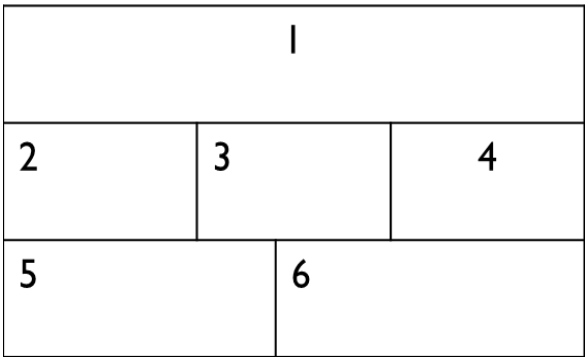
```
    box-shadow: rgba(99, 99, 99, 0.2) 0px 2px 8px 0px;
    border: none;
   }
   .userlogin-form form input:focus {
    outline: none;
   }
   .userlogin-form form .submitBtn {
    padding: 12px;
    width: 100%;
    background-color: #141e46;
    border: none;
    color: #fff5e0;
    border-radius: 6px;
    margin-top: 12px;
    cursor: pointer;
   }
  </style>
</head>
<body>
  <div class="userlogin-form">
   <h1>User Registration form</h1>
   <form
    id="registration-form"
    action="
    "
   >
    <div class="input-group">
     <label for="username">Username :</label>
     <input type="text" id="username" name="username" required />
    </div>
    <div class="input-group">
      <label for="email">Email:</label>
      <input type="email" id="email" name="email" required />
     </div>
     <div class="input-group">
      <label for="password">Password:</label>
      <input type="password" id="password" name="password" required />
     </div>
     <button class="submitBtn" type="submit">Register</button>
    </form>
    <hr />
   </div>
  </body>
</html>
```

Output:


Cisco 220 Series Sm...
Cisco CLI Switch Co...
A Practice Solutions
Code in Place
Coursera | Online C...
All Bookmarks
User Registration form
Username :
Email:
Password:
Register

Lab 2

Task 1: Using CSS and HTML design following layout



Theory

Designing layouts with CSS and HTML involves the strategic arrangement of elements on a webpage for optimal presentation and user experience. In this task, the layout is constructed using a combination of HTML table elements and CSS styling. Tables provide a structured grid, while CSS is applied to control the visual aspects like spacing, colors, and positioning. This approach allows for the creation of diverse layouts, showcasing the flexibility of CSS in organizing content. Such practices are crucial in web development to achieve aesthetically pleasing and responsive designs that cater to various screen sizes and devices.

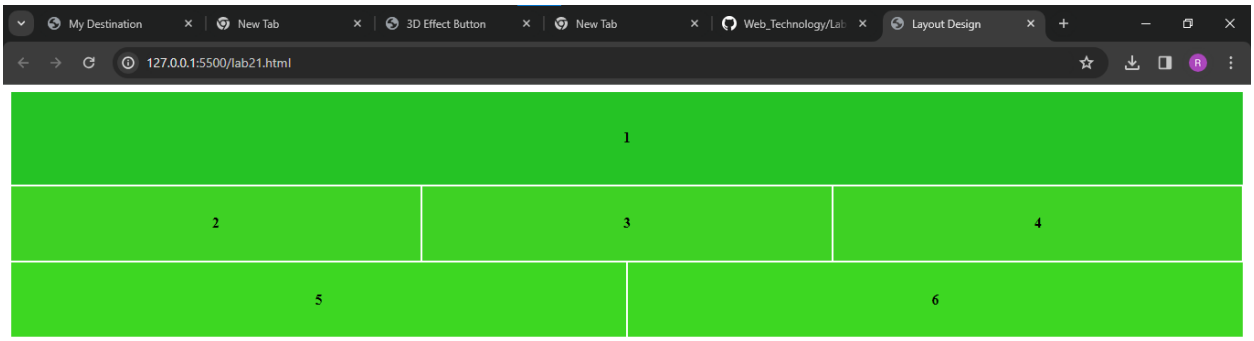
Code

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

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

    <title>Table Structure</title>

    <style>

      table {

border-collapse: collapse;

        width: 20%;
```

```
    margin: 20px auto;
   }
   td {
    border: 1px solid #000000;
    padding: 8px;
    text-align: center;
   }
  </style>
</head>
<body>
  <table>
   <tr>
    <td colspan="3">1</td>
   </tr>
   <tr>
    <td>2</td>
    <td>3</td>
    <td>4</td>
   </tr>
   <tr>
    <td colspan="2">5</td>
    <td colspan="2">6</td>
   </tr>
```

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

Output:



Task 2: Design your class routine using appropriate css.

Theory

Creating an effective class routine involves the skillful integration of CSS and HTML tables. CSS properties such as borders, padding, and background colors play a pivotal role in enhancing the visual appeal and functionality of the routine. Borders provide clarity and structure, while padding ensures well-defined spacing, preventing visual clutter. Background colors contribute to a positive learning environment. This harmonious blend of CSS and HTML elements results in an organized, engaging, and user-friendly class routine, facilitating seamless communication of essential information.

Code

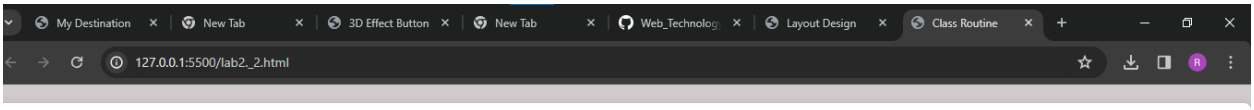
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Class Routine</title>
    <link rel="stylesheet" href="styles.css" />
    <style>
      body {
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        background-color: #d1cbcb;
      }
      .class-routine {
        width: auto;
        margin: 20px auto;
        background-color: #fff;
        padding: 20px;
        border-radius: 8px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
```

```
    }
    h1 {
      text-align: center;
      color: #333;
    }
    table {
      width: 100%;
      border-collapse: collapse;
      margin-top: 20px;
    }
    th,
    td {
      padding: 12px;
      text-align: left;
      border: 1px solid #0c0c0c;
    }
    th {
      background-color: #f2f2f2;
    }
    tr:hover {
      background-color: #f5f5f5;
    }
  </style>
</head>
<body>
  <div class="class-routine">
    <h1>BScCSIT 5th Semester Class Routine</h1>
    <table>
      <tr>
        <th>Time</th>
```

```
    <th>Sunday</th>
    <th>Tuesday</th>
    <th>Wednesday</th>
    <th>Thursday</th>
    <th>Friday</th>
  </tr>
  <tr>
    <td>7:00 AM - 9:00 AM</td>
    <td>DAA (Design and Analysis of Algorithms)</td>
    <td>Web Technology</td>
    <td>Simulation and Modeling</td>
    <td>Cryptography</td>
    <td>EPS</td>
  </tr>
  <tr>
    <td>10:00 AM - 11:00 AM</td>
    <td>Cryptography</td>
    <td>SAD (System Analysis and Design)</td>
    <td>DAA (Design and Analysis of Algorithms)</td>
    <td>Web Technology</td>
    <td>Break</td>
  </tr>
  <tr>
    <td>11:00 AM - 12:00 PM</td>
    <td>Image Processing</td>
    <td>Simulation and Modeling</td>
    <td>Web Technology</td>
    <td>maths</td>
    <td>Practical class</td>
  </tr>
```

```
      <tr>
        <td>12:00 PM - 1:00 PM</td>
        <td>Lunch</td>
        <td>Lunch</td>
        <td>Lunch</td>
        <td>Lunch</td>
        <td>Lunch</td>
      </tr>
      <tr>
        <td>1:00 PM - 2:00 PM</td>
        <td>Image Processing</td>
        <td>DAA (Design and Analysis of Algorithms)</td>
        <td>Multimedia</td>
        <td>Simulation and Modeling</td>
        <td>Web Technology</td>
      </tr>
      <tr>
          <td>2:00 PM - 3:00 PM</td>
          <td>Simulation and Modeling</td>
          <td>web technology</td>
          <td>Multimedia</td>
          <td>DAA (Design and Analysis of Algorithms)</td>
          <td>SAD (System Analysis and Design)</td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

Output



BScCSIT 5th Semester Class Routine

Time	Sunday	Tuesday	Wednesday	Thursday	Friday
7:00 AM - 9:00 AM	DAA (Design and Analysis of Algorithms)	Web Technology	Simulation and Modeling	Cryptography	EPS
10:00 AM - 11:00 AM	Cryptography	SAD (System Analysis and Design)	DAA (Design and Analysis of Algorithms)	Web Technology	Break
11:00 AM - 12:00 PM	Image Processing	Simulation and Modeling	Web Technology	maths	Practical class
12:00 PM - 1:00 PM	Lunch	Lunch	Lunch	Lunch	Lunch
1:00 PM - 2:00 PM	Image Processing	DAA (Design and Analysis of Algorithms)	Multimedia	Simulation and Modeling	Web Technology
2:00 PM - 3:00 PM	Simulation and Modeling	web technology	Multimedia	DAA (Design and Analysis of Algorithms)	SAD (System Analysis and Design)

Task 3: Write a code for the position relative, absolute, fixed and make html page using css position

Theory

CSS positioning is a powerful feature that allows developers to precisely control the placement of elements on a webpage. This lab explores the relative, absolute, and fixed positioning properties. Elements with relative positioning are adjusted relative to their normal positions, absolute positioning allows precise placement within a container, and fixed positioning ensures elements stay fixed relative to the viewport. These techniques provide flexibility in crafting intricate and responsive layouts.

Code

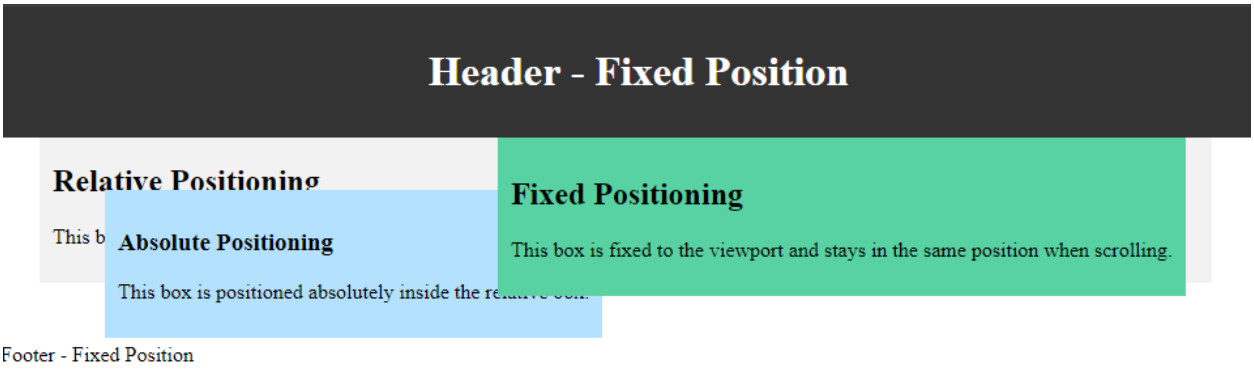
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Positioning Example</title>
    <style>
      body {
        margin: 0;
        padding: 0;
      }
      header {
        background-color: #333;
        color: #fff;
        text-align: center;
        padding: 10px;
        position: fixed;
        top: 0;
        left: 0;
        width: 100%;
```

```
  z-index: 1000;
}
section {
  margin-top: 60px;
  padding: 20px;
}
.relative-box {
  position: relative;
  background-color: #f2f2f2;
  padding: 10px;
  margin: 10px;
}
.absolute-box {
  position: absolute;
  top: 50px;
  left: 50px;
  background-color: #b3e0ff;
  padding: 10px;
}
.fixed-box {
  position: fixed;
  top: 100px;
  right: 50px;
  background-color: #ffc0cb;
  padding: 10px;
footer {
  background-color: #333;
  color: #fff;
  text-align: center;
  padding: 10px;
```

```
      position: fixed;
      bottom: 0;
      left: 0;
      width: 100%;
    }
  </style>
</head>
<body>
  <header>
    <h1>Header - Fixed Position</h1>
  </header>
  <section>
    <div class="relative-box">
      <h2>Relative Positioning</h2>
      <p>This box is positioned relative to its normal position.</p>
      <div class="absolute-box">
        <h3>Absolute Positioning</h3>
        <p>This box is positioned absolutely inside the relative box.</p>
      </div>
    </div>
    <div class="fixed-box">
      <h2>Fixed Positioning</h2>
      <p>
        This box is fixed to the viewport and stays in the same position when
        scrolling.
      </p>
    </div>
  </section>
  <footer>
    <p>Footer - Fixed Position</p>
```

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

Output



Task 4 : Using CSS design attractive 3D effect buttons

Theory

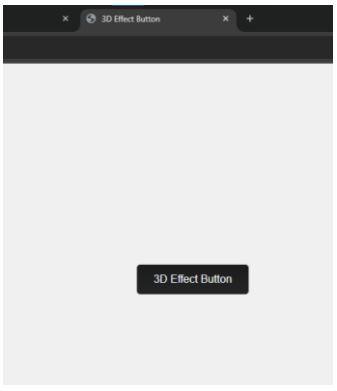
Creating visually appealing buttons is a common task in web development. In this lab, CSS is utilized to design buttons with 3D effects. Techniques such as gradients, shadows, and border styles contribute to the creation of attractive and interactive buttons. This enhances the overall user interface and encourages user engagement. more long

Code

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>3D Effect Button</title>
    <style>
      body {
        display: flex;
        align-items: center;
        justify-content: center;
        height: 100vh;
        margin: 0;
        background-color: #f0f0f0;
      }
      .button {
        padding: 12px 24px;
        font-size: 16px;
        text-align: center;
        text-decoration: none;
        cursor: pointer;
```

```
      color: #fff;
      background: linear-gradient(180deg, #4caf50 0%, #45a049 100%);
      border: 1px solid #4caf50;
      border-radius: 5px;
      transition: transform 0.3s;
    }
    .button:hover {
      transform: translateY(1px);
      box-shadow: 0 6px 12px rgb(34, 169, 52);
    }
    .button:active {
      transform: translateY(0);
    }
  </style>
 </head>
 <body>
  <button class="button">3D Effect Button</a>
 </body>
</html>
```

Output



Lab 3

Task : Create a user registration form having fields user name, email, phone, password and validate the form.

***username required**

***email required, proper format**

***phone optional, number format**

***password required, min 8 characters**

Theory

User registration forms are the entry point for users into web applications. This simplified process includes fields for username, email, optional phone, and a mandatory password. Client-side validation, powered by JavaScript, ensures immediate feedback and data accuracy.

The username and email fields are marked as required, emphasizing their necessity for successful registration. While the phone field is optional, it only accepts numerical input if provided. Passwords, a critical security aspect, must be a minimum of 8 characters.

The form structure is straightforward, featuring clear labels for each field and an onsubmit attribute triggering client-side validation. This integration ensures a seamless user experience, combining user-friendliness with data integrity in the user registration journey.

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>

      body {

        background-color: #f4f4f4;

        margin: 0;

        padding: 0;
```

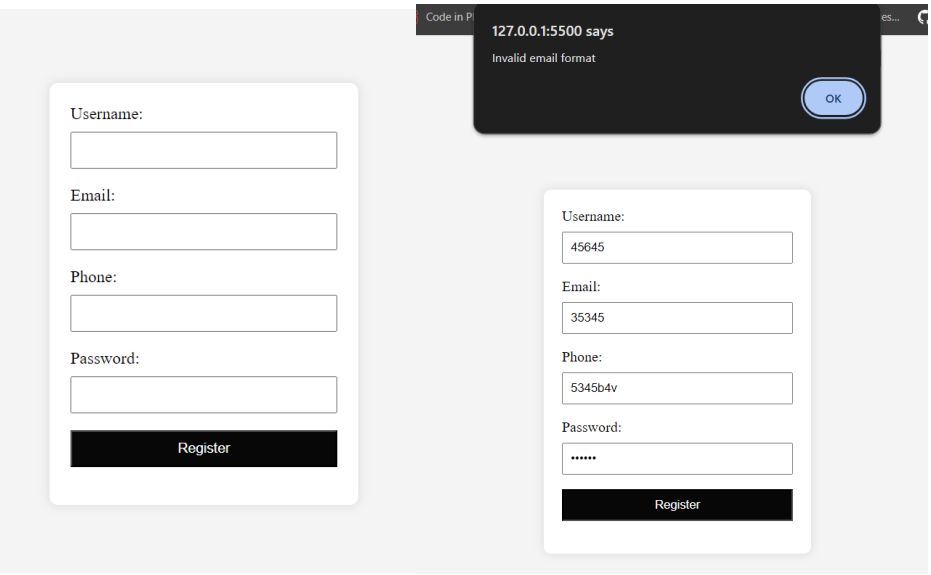
```
  display: flex;
  align-items: center;
  justify-content: center;
  height: 100vh;
}
form {
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
label {
  display: block;
  margin-bottom: 8px;
}
input {
  width: 100%;
  padding: 8px;
  margin-bottom: 16px;
  box-sizing: border-box;
}
input[type="submit"] {
  background-color: #4caf50;
  color: #fff;
  cursor: pointer;
```

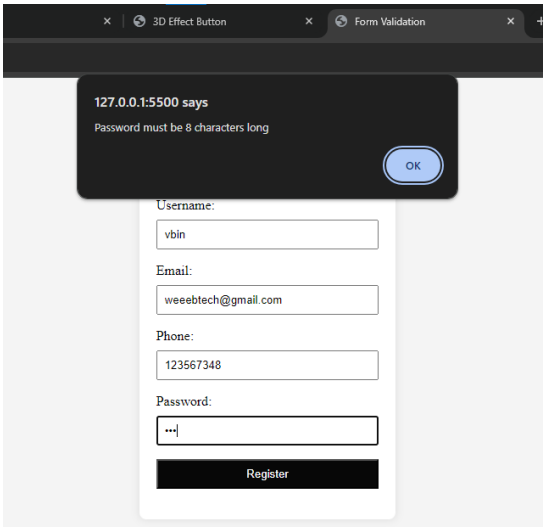
```
    }
  </style>
</head>
<body>
  <form action="" onsubmit="return formSubmit()">
    <label for="username">Username:</label>
    <input type="text" id="username" />

    <label for="email">Email:</label>
    <input type="text" id="email" />

    <label for="phonenumber">Phone:</label>
    <input type="text" id="phonenumber" />

    <label for="password">Password:</label>
    <input type="password" id="password" />

    <input type="submit" value="Register" />
  </form>
  <script>
    function formSubmit() {
      var username = document.getElementById("username").value;
      var email = document.getElementById("email").value;
      var phone = document.getElementById("phonenumber").value;
      var password = document.getElementById("password").value;
```

```
if (
  username === "" ||
  email === "" ||
  phone === "" ||
  password === ""
) {
  alert("Please fill up all data");
  return false;
}
var emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
if (!emailRegex.test(email)) {
var password = document.getElementById("password").value;
if (
  username === "" ||
  email === "" ||
  phone === "" ||
  password === ""
) {
  alert("Please fill up all data");
  return false;
}
var emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
if (!emailRegex.test(email)) {
  alert("Invalid email format");
  return false;
```

```
    }
    var phoneRegex = /^\d*$/;
    if (!phoneRegex.test(phone)) {
      alert("Invalid phone number format");
      return false;
    }
    if (password.length < 8) {
      alert("Password must be 8 characters long");
      return false;
    }
    return true;
  }
  </script>
</body>
</html>
```

Output




3D Effect Button
Form Validation
127.0.0.1:5500 says
Password must be 8 characters long
OK
Username:
vbin
Email:
weeebtech@gmail.com
Phone:
123567348
Password:
Register

Lab 4

Task: Develop a simple website that checks the validity of the user login. Assume that the data is already in the database. Use Client Site script to check the empty values during login

Theory

Client-side user login validation, implemented through JavaScript, enhances the web experience by providing instant feedback to users and reducing server requests. It ensures essential fields are not left empty, improving responsiveness and minimizing wait times. However, it should complement, not replace, server-side validation for comprehensive security. Despite its benefits, client-side validation has limitations, such as code exposure. In summary, it significantly improves user experience but should be integrated thoughtfully with server-side validation for a robust and secure web environment.

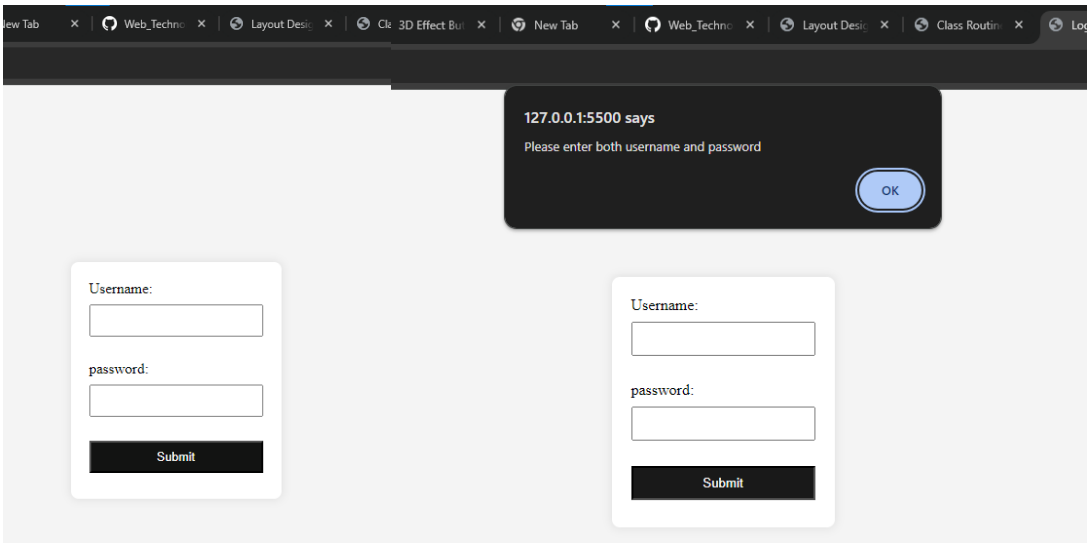
Code

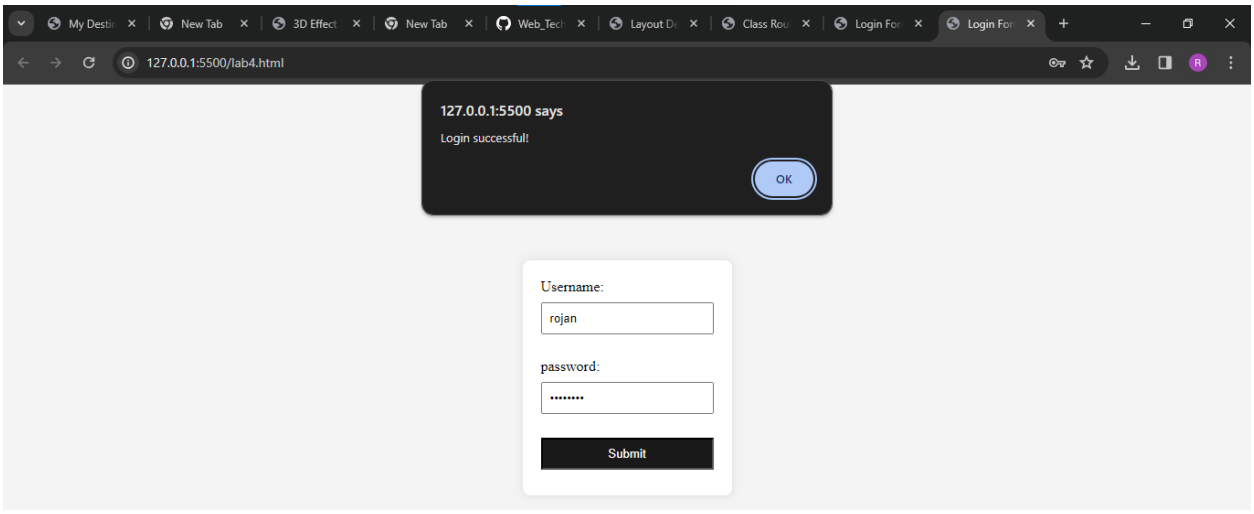
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Login Form</title>
    <style>
      body {
        background-color: #f4f4f4;
        margin: 0;
        padding: 0;
        display: flex;
        align-items: center;
        justify-content: center;
        height: 100vh;
      }

      form {
```

```
      background-color: #fff;
      padding: 20px;
      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: 8px;
      box-sizing: border-box;
    }

    input[type="submit"] {
      background-color: #121312;
      color: #fff;
      cursor: pointer;
   }

input[type="submit"]:hover {
  background-color: #191919;
}
</style>
</head>
<body>
```

```
<form onsubmit="return formvalidation()">
<label for="username">Username:</label>
<input type="text" id="username" /><br /><br />
<label for="password">password:</label>
<input type="password" id="password" /><br /><br />
<input type="submit" />
</form>
</body>

<script>
function formvalidation(event) {
var username = document.getElementById("username").value;
var password = document.getElementById("password").value;
if (username === "" || password === "") {
  alert("Please enter both username and password");
} else {
  var dbUsername = "rojan";
  var dbpassword = "hello123";
       if (username === dbUsername && password === dbpassword) {
         alert("Login successful!");
       } else {
         alert("Invalid username or password");
       }
     }
   }
  </script>
</html>
```

Output:


127.0.0.1:5500 says
Please enter both username and password
OK
Username:
password:
Submit
Username:
password:
Submit


127.0.0.1:5500/lab4.html
127.0.0.1:5500 says
Login successful!
OK
Username:
rojan
password:
Submit

Lab 5

Task 1: Create a database Student and create a Table (Table 1) in MySQL having following:

Column Name	ID	Firstname	Lastname	Gender	semester	Symbol. no	Batch
Data Type	Int	Varchar	Varchar	Boolean (Male/Female)	int	varchar	int

Theory

The initiation of a MySQL database named "Student" introduces a systematic approach to managing academic information. The pivotal creation of "Table1" demonstrates meticulous planning, incorporating key columns—ID, FirstName, LastName, Gender, Semester, Symbol, and Batch—each assigned an appropriate data type. This design ensures efficient storage and retrieval of student records, adhering to principles of normalization. The boolean representation for gender simplifies data clarity, providing a concise and binary view of this attribute. This strategic database design forms the foundation for effective data management in subsequent lab activities.

For database creation

Code

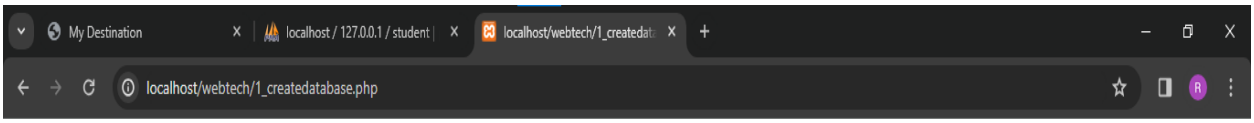
```php
<?php
$servername = "localhost";
$username = "root";
$password = "";

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

// Create database
$sql = "CREATE DATABASE student";
if (mysqli_query($conn, $sql)) {
```

```
    echo "Database created successfully";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

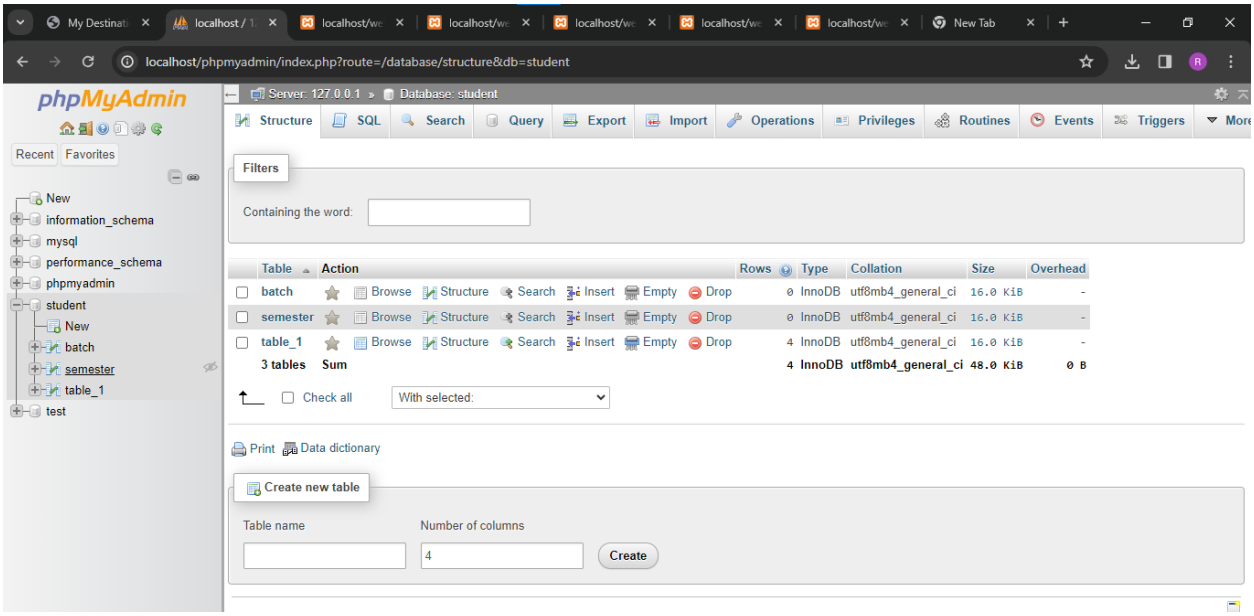
Code for table

```
<?php

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "student";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
// sql to create table
$sql = "CREATE TABLE table_1 (
ID INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
Firstname VARCHAR(30) NOT NULL,
Lastname VARCHAR(30) NOT NULL,
Gender BOOLEAN NOT NULL,
Semester INT(2) UNSIGNED NOT NULL,
SymbolNumber VARCHAR(30) NOT NULL,
Batch INT(2) UNSIGNED NOT NULL
)";
if (mysqli_query($conn, $sql)) {
    echo "Table student created successfully";
} else {
    echo "Error creating table: ". mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Output



Database created successfully



Task 2: Create Separate Tables for Class and Batch containing ID,name

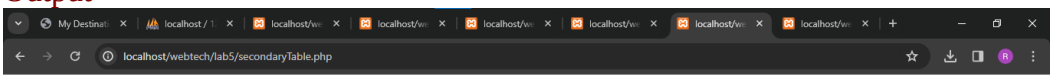
Theory

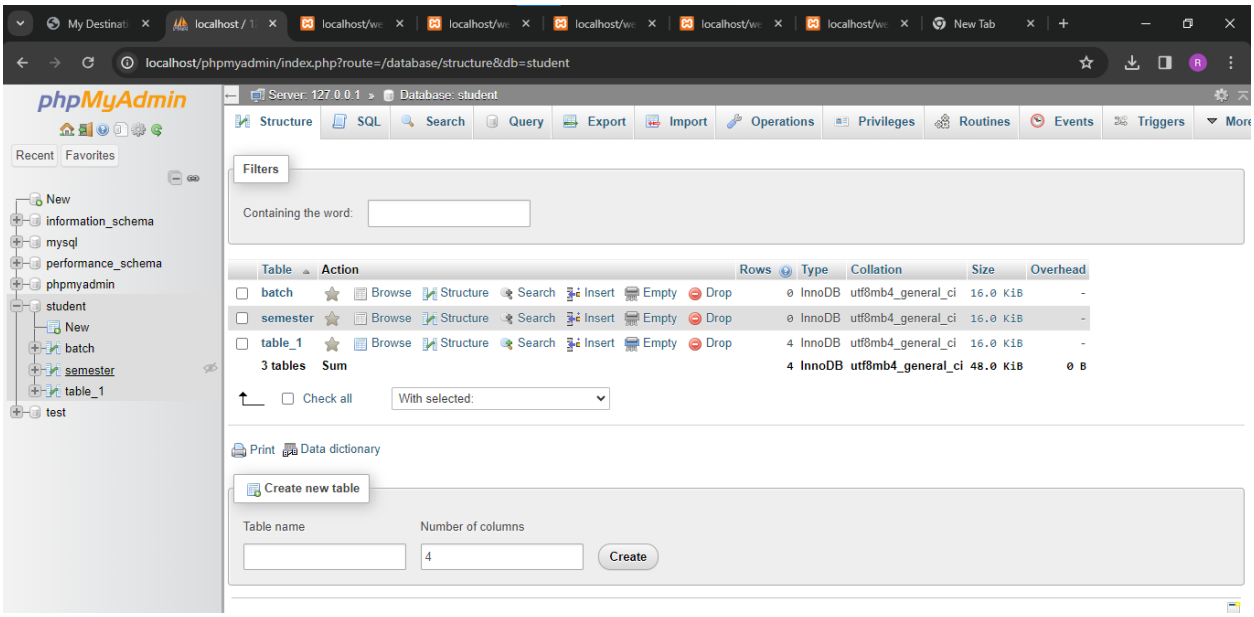
A hallmark of database normalization and foresight, two supplementary tables— "Class" and "Batch"—are forged to bring order and flexibility. Each table features columns for ID and Name, creating a modular structure. The "Class" table becomes the curator of academic classes, while "Batch" diligently manages batches. This meticulous approach anticipates the evolving needs of academic organization, providing a scalable solution for a system in perpetual growth.

Code

```php
<?php
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "student";
    $conn = mysqli_connect($servername, $username, $password, $dbname);
    if (!$conn) {
       die("Connection failed: ".mysqli_connect_error());
    }
    // sql to create table
    $sql = "CREATE TABLE semester (
       ID INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
       Name VARCHAR(30) NOT NULL
    )";
    if (mysqli_query($conn, $sql)) {
       echo "Table class created successfully";
    } else {
       echo "Error creating table: ".mysqli_error($conn);
    }
    $sql = "CREATE TABLE batch (
       ID INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
       Name VARCHAR(30) NOT NULL
    )";
    if (mysqli_query($conn, $sql)) {
       echo "Table batch created successfully";
    } else {
       echo "Error creating table: ".mysqli_error($conn);
    }
    mysqli_close($conn);
    ?>
```

Output


localhost/webtech/lab5/secondaryTable.php
Table class created successfullyTable batch created successfully


localhost/phpmyadmin/index.php?route=/database/structure&db=student
phpMyAdmin
Recent
Favorites
New
information_schema
mysql
performance_schema
phpmyadmin
student
New
batch
semester
table_1
test
Server: 127.0.0.1 » Database: student
Structure
SQL
Search
Query
Export
Import
Operations
Privileges
Routines
Events
Triggers
Filters
Containing the word:
Table
Action
Rows
Type
Collation
Size
Overhead
batch Browse Structure Search Insert Empty Drop 0 InnoDB utf8mb4_general_ci 16.0 KiB -
semester Browse Structure Search Insert Empty Drop 0 InnoDB utf8mb4_general_ci 16.0 KiB -
table_1 Browse Structure Search Insert Empty Drop 4 InnoDB utf8mb4_general_ci 16.0 KiB -
3 tables Sum 4 InnoDB utf8mb4_general_ci 48.0 KiB 0 B
Check all
With selected:
Print
Data dictionary
Create new table
Table name
Number of columns
4
Create

Task 3: Create a Simple form (Form 1) in PHP having FirstName, Lastname, Gender (Radio Button), Class (dropdown), Roll Number and Batch (dropdown)

Theory

The creation of Form 1 in PHP transcends mere data entry; it is a crafted interface facilitating meaningful user interaction. Fields spanning FirstName to Batch are meticulously arranged, showcasing the dynamic capabilities of PHP in content generation. The design of this form doesn't just collect data; it acts as a portal, inviting users to contribute to the Student database. This emphasis on thoughtful design underscores the commitment to a user-centric approach.

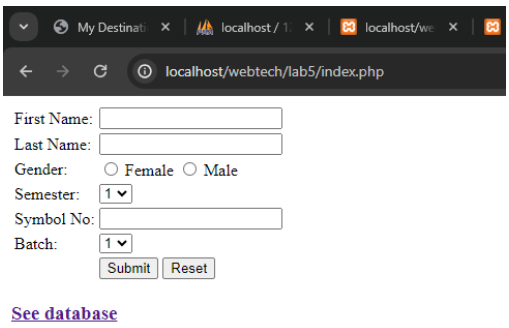
Code

```
<!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></title>
</head>
<body>
  <form method="post" action="insert.php" onsubmit="return validateForm()">
    <table>
      <tr>
        <td><label for="fname">First Name:</label></td>
        <td><input type="text" id="fname" name="firstname" required></td>
      </tr>

      <tr>
        <td><label for="lname">Last Name:</label></td>
        <td><input type="text" id="lname" name="lastname" required></td>
      </tr>

      <tr>
        <td><label>Gender:</label></td>
        <td>
          <input type="radio" id="female" value="1" name="gender" required> Female
          <input type="radio" id="male" value="0" name="gender"> Male
        </td>
      </tr>

      <tr>
        <td><label for="semester">Semester:</label></td>
        <td>
          <select name="semester" id="semester">
            <option value="1">1</option>
```

```
                <option value="2">2</option>
                <option value="3">3</option>
                <option value="4">4</option>
                <option value="5">5</option>
                <option value="6">6</option>
                <option value="7">7</option>
                <option value="8">8</option>
            </select>
        </td>
    </tr>

    <tr>
        <td><label for="symbol">Symbol No:</label></td>
        <td><input type="text" id="symbol" name="symbol" required></td>
    </tr>

    <tr>
        <td><label for="batch">Batch:</label></td>
        <td>
            <select name="batch" id="batch">
                <option value="1">1</option>
                <option value="2">2</option>
                <option value="3">3</option>
                <option value="4">4</option>
            </select>
        </td>
    </tr>

    <tr>
        <td></td>
        <td>
            <input type="submit" value="Submit">
            <input type="reset">
        </td>
    </tr>
</table>
</form>
<script>
function validateForm() {
    // Get values from the form
    const firstname = document.getElementById('fname').value;
    const lastname = document.getElementById('lname').value;
    const gender = document.querySelector('input[name="gender"]:checked');
    const selectedSemester = document.getElementById('semester').value;
    const symbolnumber = document.getElementById('symbol').value;
```

```
        const selectedBatch = document.getElementById('batch').value;

        // Detailed validation
        if (firstname === "") {
            alert("Please enter your First Name");
            return false;
        }
        if (lastname === "") {
            alert("Please enter your Last Name");
            return false;
        }
        if (!gender) {
            alert("Please select your Gender");
            return false;
        }
        if (selectedSemester === "") {
            alert("Please select your Class");
            return false;
        }
        if (symbolnumber === "") {
            alert("Please enter your Roll Number");
            return false;
        }
        if (selectedBatch === "") {
            alert("Please select your Batch");
            return false;
        }

        return true;
    }
    </script>
    <a href="list.php"><h3>See database</h3></a>
</body>
</html>
```

Output



Task 4 : Use Client Side required validation in the Form 1

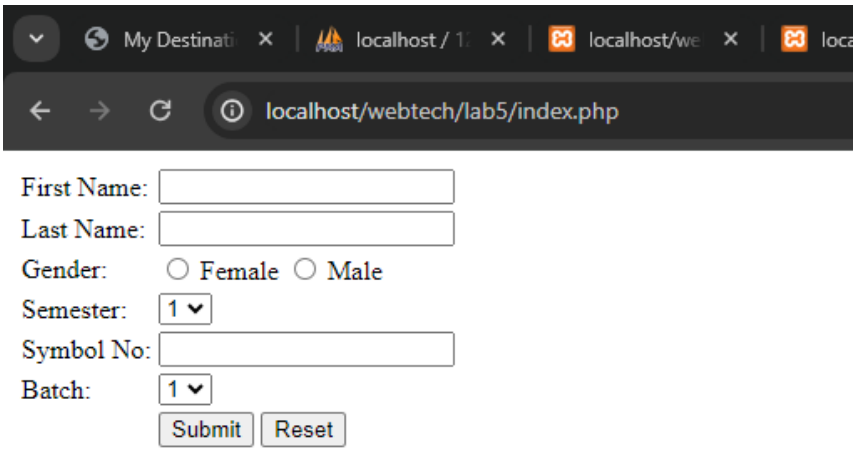
Code

```
<!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></title>
</head>
<body>
   <form method="post" action="insert.php" onsubmit="return validateForm()">
      <table>
         <tr>
            <td><label for="fname">First Name:</label></td>
            <td><input type="text" id="fname" name="firstname" required></td>
         </tr>

         <tr>
            <td><label for="lname">Last Name:</label></td>
            <td><input type="text" id="lname" name="lastname" required></td>
         </tr>

         <tr>
            <td><label>Gender:</label></td>
            <td>
               <input type="radio" id="female" value="1" name="gender" required> Female
               <input type="radio" id="male" value="0" name="gender"> Male
            </td>
         </tr>

         <tr>
            <td><label for="semester">Semester:</label></td>
            <td>
               <select name="semester" id="semester">
                  <option value="1">1</option>
                  <option value="2">2</option>
                  <option value="3">3</option>
                  <option value="4">4</option>
                  <option value="5">5</option>
                  <option value="6">6</option>
                  <option value="7">7</option>
                  <option value="8">8</option>
```

```
            </select>
        </td>
    </tr>

    <tr>
        <td><label for="symbol">Symbol No:</label></td>
        <td><input type="text" id="symbol" name="symbol" required></td>
    </tr>

    <tr>
        <td><label for="batch">Batch:</label></td>
        <td>
            <select name="batch" id="batch">
                <option value="1">1</option>
                <option value="2">2</option>
                <option value="3">3</option>
                <option value="4">4</option>
            </select>
        </td>
    </tr>

    <tr>
        <td></td>
        <td>
            <input type="submit" value="Submit">
            <input type="reset">
        </td>
    </tr>
</table>
</form>
<script>
function validateForm() {
    // Get values from the form
    const firstname = document.getElementById('fname').value;
    const lastname = document.getElementById('lname').value;
    const gender = document.querySelector('input[name="gender"]:checked');
    const selectedSemester = document.getElementById('semester').value;
    const symbolnumber = document.getElementById('symbol').value;
    const selectedBatch = document.getElementById('batch').value;

    // Detailed validation
    if (firstname === "") {
        alert("Please enter your First Name");
        return false;
    }
```

```
        if (lastname === "") {
            alert("Please enter your Last Name");
            return false;
        }
        if (!gender) {
            alert("Please select your Gender");
            return false;
        }
        if (selectedSemester === "") {
            alert("Please select your Class");
            return false;
        }
        if (symbolnumber === "") {
            alert("Please enter your Roll Number");
            return false;
        }
        if (selectedBatch === "") {
            alert("Please select your Batch");
            return false;
        }

        return true;
    }
    </script>
    <a href="list.php"><h3>See database</h3></a>
</body>
</html>
```

Output



Task 5 : Connect Database and Table 1 and insert the data filled in Form 1

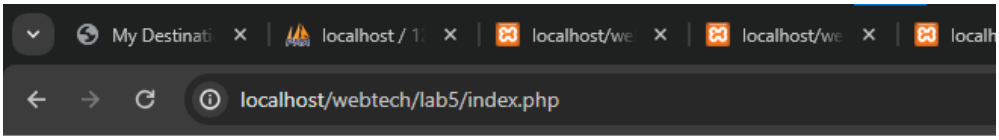
Theory

The orchestration of PHP takes center stage in establishing a secure and efficient bridge to the MySQL database. This connection, far from a mere conduit, becomes the conduit through which user-inputted data elegantly integrates into "Table1." The incorporation of prepared statements in PHP is not merely a security measure; it is a shield against SQL injection attacks, fortifying the system's defenses against potential vulnerabilities. This robust integration is emblematic of a commitment to data security and the seamless management of information.

Code

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "student";
$conn = mysqli_connect($servername, $username, $password, $dbname);

if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

$firstname = $_REQUEST['firstname'];
$lastname = $_REQUEST['lastname'];
$gender = $_REQUEST['gender'];
$semester = $_REQUEST['semester'];
$symbol = $_REQUEST['symbol'];
$batch = $_REQUEST['batch'];

$sql = "INSERT INTO table_1 (Firstname, Lastname, Gender, Semester, SymbolNumber, Batch)
VALUES ('$firstname', '$lastname', '$gender', '$semester', '$symbol', '$batch')";

if (mysqli_query($conn, $sql)) {
   header('Location: index.php');
} else {
   echo "ERROR: Hush! Sorry $sql." . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

Output



First Name: Rojan
Last Name: Gurung
Gender: Female Male
Semester: 1
Symbol No: 26059
Batch: 1
Submit Reset

See database

Task 6 : Connect Database and Table 1 and insert the data filled in Form 1

Theory

The orchestration of PHP takes center stage in establishing a secure and efficient bridge to the MySQL database. This connection, far from a mere conduit, becomes the conduit through which user-inputted data elegantly integrates into "Table1." The incorporation of prepared statements in PHP is not merely a security measure; it is a shield against SQL injection attacks, fortifying the system's defenses against potential vulnerabilities. This robust integration is emblematic of a commitment to data security and the seamless management of information.

Code

```
<?php
$username = "root";
$password = "";
$database = "student";
$mysqli = new mysqli("localhost", $username, $password, $database);
$query = "SELECT * FROM table_1";
echo "<b> <center>Database Output</center> </b> <br> <br>";
if ($result = $mysqli->query($query)) {
    echo '<table style="width:100%">';
    echo '<tr style="font-size: 20px; font-weight:bold;">';
    echo '<td>ID</td>';
    echo '<td>Firstname</td>';
    echo '<td>Lastname</td>';
    echo '<td>Gender</td>';
    echo '<td>Semester</td>';
    echo '<td>SymbolNumber</td>';
    echo '<td>Batch</td>';
    echo '</tr>';
    while ($row = $result->fetch_assoc()) {
        $id = $row["ID"];
        $firstname = $row["Firstname"];
        $lastname = $row["Lastname"];
        $gender = $row["Gender"];
        $semester = $row["Semester"];
        $symbol = $row["SymbolNumber"];
        $batch = $row["Batch"];
        echo '<tr style="font-size:18px">';
        echo '<td>' . $id . '</td>';
        echo '<td>' . $firstname . '</td>';
        echo '<td>' . $lastname . '</td>';
        echo '<td>' . $gender . '</td>';
        echo '<td>' . $semester. '</td>';
        echo '<td>' . $symbol. '</td>';
        echo '<td>' . $batch . '</td>';
```

```
        echo '</tr>';
    }
    echo '</table>';
    /* free result set */
    $result->free();
}
?>
```

Database Output

ID	Firstname	Lastname	Gender	Semester	SymbolNumber	Batch
1	Rojan	Gurung	0	5	26059	1
2	Rojan	Gurung	0	1	26059	1
3	Rojan	Gurung	0	5	26059	1
4	Rojan	Gurung	0	1	26059	1