

# Intro To WebTech Lab



Intro To WebTech Lab,  
Department of Computer Science,  
School of Technology,  
Pandit Deendayal Energy University

By:

Tirth Shah,

22BCP230

Under the guidance of:

Komal Singh, Department of Computer Science, School of  
Technology, Pandit Deendayal Energy University

# Certificate



This is to certify that **Tirth Shah**, student of **G6-Div3 CSE'26** with  
enrolment number **22BCP230** has satisfactorily completed his work  
in **Intro To WebTech Lab** under the guidance of **Komal Singh**.

---

Lab Instructor

---

Head of the department

# Practical-1

Q1. Make a webpage with basic HTML and CSS

A1. **Explanation of the code used:**

## **HTML Tags:**

- **<h5>, <h6>:** Define headings of different levels (smaller than <h1>).
- **<p>:** Defines a paragraph of text.
- **<a>:** Creates a hyperlink to another webpage or location.
- **<img>:** Embeds an image in the document.
- **<strong>:** Displays text with strong importance (usually bold).
- **<i>:** Displays text in italics for emphasis.
- **<u>:** Underlines the enclosed text.
- **<b>:** Makes text bold.
- **<em>:** Emphasizes text (usually italics).
- **<mark>:** Highlights or marks text.
- **<small>:** Reduces the size of the enclosed text.
- **<del>:** Strikethrough text to indicate deletion.
- **<ins>:** Underlines text to indicate insertion.
- **<sub>:** Subscript text (smaller text below the baseline).
- **<sup>:** Superscript text (smaller text above the baseline).
- **<q>:** Adds inline quotations.
- **<blockquote>:** Defines a long block of quoted text.
- **<abbr>:** Defines an abbreviation with a title on hover.
- **<address>:** Defines contact information.
- **<table>:** Creates a table to organize data.
- **<tr>:** Defines a table row.
- **<th>:** Defines a header cell in a table.
- **<td>:** Defines a standard data cell in a table.

## CSS Properties:

- **img { width: 150px; }:** Sets the image width to 150 pixels.
- **table { border-collapse: collapse; text-align: center; }:** Collapses table borders into a single border and centers the table content.
- **table, th, td { border: 1px solid black; }:** Applies a 1px solid black border to the table, table headers (th), and data cells (td).
- **td, th { padding: 3rem; width: max-content; height: max-content; }:** Adds padding of 3 rem and sets the width and height of cells to adjust based on content.

## Code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Lecture1</title>
    <style>
      img {
        width: 150px;
      }
      table {
        border-collapse: collapse;
        text-align: center;
      }
      table,
      th,
      td {
        border: 1px solid black;
      }
      td,
      th {
        padding: 3rem;
        width: max-content;
        height: max-content;
      }
    </style>
    <link rel="icon" href="/20190503_160252.jpg" type="image/x-icon" />
  </head>
  <body>
    <h5>This is h5 heading tag</h5>
    <h6>This is h6 heading tag</h6>
    <p>This is a paragraph tag</p>
    <a href="https://www.google.com">This is a link to Google</a><br /><br />
    <br /><br />
    <strong>This is a text formatting tag - Strong</strong><br /><br />
    <i>This is a text formatting tag - Italics</i><br /><br />
```

```

<u>This is a text formatting tag - Underline</u><br /><br />
<b>This is a text formatting tag - Bold</b><br /><br />
<em>This is a text formatting tag - Emphasis</em><br /><br />
<mark>This is a text formatting tag - Mark</mark><br /><br />
<small>This is a text formatting tag - Small</small><br /><br />
<del>This is a text formatting tag - Deleted</del><br /><br />
<ins>This is a text formatting tag - Inserted</ins><br /><br />
<span>To show that</span>
><sub> this is a text formatting tag - Subscript</sub><br /><br />
<span>To show that</span>
><sup> this is a text formatting tag - Superscript</sup>
<p>Rishabh said: <q>Durg AC On Karna</q></p>
<p>Original Bitcoin Paper stated:</p>
<blockquote cite="https://bitcoin.org/bitcoin.pdf">
  A purely peer-to-peer version of electronic cash would allow online
  payments to be sent directly from one party to another without going
  through a financial institution
</blockquote>
<p>
  Today we are learning
  <abbr title="Hyper Text Markup Language">HTML</abbr>.
</p>
<p>Written at:</p>
<address>
  PDEU, Knowledge Corridor,<br />Raysan, Gandhinagar,<br />Gujarat, India
</address>
<br /><br />
<table>
  <tr>
    <th>Roll</th>
    <th>Name</th>
    <th>Fav Field</th>
  </tr>
  <tr>
    <td>22BCP222</td>
    <td>Mahimna</td>
    <td>Cooking</td>
  </tr>
  <tr>
    <td>22BCP230</td>
    <td>Tirth</td>
    <td>Cloud Computing</td>
  </tr>
  <tr>
    <td>22BCP233</td>
    <td>Rishabh</td>
    <td>Music</td>
  </tr>
  <tr>
    <td>22BCP221</td>
    <td>Durg</td>
  </tr>

```

```

        <td>Gaming</td>
    </tr>
    <tr>
        <td>22BCP232</td>
        <td>Rudra</td>
        <td rowspan="2">Web Development</td>
    </tr>
    <tr>
        <td>22BCP217</td>
        <td>Vasu</td>
    </tr>
</table>
<p>Table to learn Rowspan and Colspan</p>
<table>
    <tr>
        <td><strong>This is a text formatting tag - Strong</strong></td>
        <td><i>This is a text formatting tag - Italics</i></td>
        <td colspan="2"><u>This is a text formatting tag - Underline</u></td>
    </tr>
    <tr>
        <td><b>This is a text formatting tag - Bold</b></td>
        <td><em>This is a text formatting tag - Emphasis</em></td>
        <td colspan="2"><mark>This is a text formatting tag - Mark</mark></td>
    </tr>
    <tr>
        <td colspan="2">
            <small>This is a text formatting tag - Small</small>
        </td>
        <td rowspan="2" colspan="2">
            <del>This is a text formatting tag - Deleted</del>
        </td>
    </tr>
    <tr>
        <td colspan="2"><ins>This is a text formatting tag - Inserted</ins></td>
    </tr>
    <tr>
        <td>
            <span>To show that</span>
            ><sub> this is a text formatting tag - Subscript</sub>
        </td>
        <td>
            <p>Rishabh said: <q>Durg AC On Karna</q></p>
        </td>
        <td rowspan="3">
            <p>Original Bitcoin Paper stated:</p>
            <blockquote cite="https://bitcoin.org/bitcoin.pdf">
                A purely peer-to-peer version of electronic cash would allow online
                payments to be sent directly from one party to another without going
                through a financial institution
            </blockquote>
        </td>
    </tr>

```

```

<td rowspan="3">
  <p>
    Today we are learning
    <abbr title="Hyper Text Markup Language">HTML</abbr>.
  </p>
</td>
</tr>
<tr>
  <td>
    <p>Written at:</p>
    <address>
      PDEU, Knowledge Corridor,<br />Raysan, Gandhinagar,<br />Gujarat,
      India
    </address>
  </td>
  <td>
    <p>Written at:</p>
    <address>
      PDEU, Knowledge Corridor,<br />Raysan, Gandhinagar,<br />Gujarat,
      India
    </address>
  </td>
</tr>
<tr>
  <td>P</td>
  <td>Q</td>
</tr>
</table>
</body>
</html>

```

# Output:

This is h5 heading tag

This is h6 heading tag

This is a paragraph tag

[This is a link to Google](#)



**This is a text formatting tag - Strong**

*This is a text formatting tag - Italics*

This is a text formatting tag - Underline

**This is a text formatting tag - Bold**

*This is a text formatting tag - Emphasis*

**This is a text formatting tag - Mark**

This is a text formatting tag - Small

~~This is a text formatting tag - Deleted~~

This is a text formatting tag - Inserted

To show that this is a text formatting tag - Subscript

To show that <sup>this is a text formatting tag - Superscript</sup>

Rishabh said: "Durg AC On Karna"

Original Bitcoin Paper stated:

A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through

Today we are learning ~~HTML~~.

Written at:

*PDEU, Knowledge Corridor,  
Raysan, Gandhinagar,  
Gujarat, India*



Roll	Name	Fav Field
22BCP222	Mahimna	Cooking
22BCP230	Tirth	Cloud Computing
22BCP233	Rishabh	Music
22BCP221	Durg	Gaming
22BCP232	Rudra	Web Development
22BCP217	Vasu	

Table to learn Rowspan and Colspan

<b>This is a text formatting tag - Strong</b>	<i>This is a text formatting tag - Italics</i>	<u>This is a text formatting tag - Underline</u>	
<b>This is a text formatting tag - Bold</b>	<i>This is a text formatting tag - Emphasis</i>	<b>This is a text formatting tag - Mark</b>	
This is a text formatting tag - Small		<del>This is a text formatting tag - Deleted</del>	
<u>This is a text formatting tag - Inserted</u>			
To show that this is a text formatting tag - Subscript	Rishabh said: "Durg AC On Karna"	Original Bitcoin Paper stated:  A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution	Today we are learn <u>HTML</u> .
Written at: <i>PDEU, Knowledge Corridor, Raysan, Gandhinagar, Gujarat, India</i>	Written at: <i>PDEU, Knowledge Corridor, Raysan, Gandhinagar, Gujarat, India</i>		

## Practical-2

Q1. Make a form with basic HTML and CSS

A1. **Explanation of code:**

### HTML Tags Used

- **<!DOCTYPE html>**: Defines the document type and version of HTML.
- **<html>**: Root element of the HTML document.
- **<head>**: Contains meta-information about the document.
- **<meta>**: Provides metadata such as character set and viewport settings.
- **<link>**: Links external CSS files.
- **<style>**: Contains internal CSS styles.
- **<title>**: Sets the title of the document.
- **<body>**: Contains the content of the document.
- **<div>**: Defines a division or section.
- **<h1>**: Defines a top-level heading.
- **<form>**: Defines an HTML form for user input.
- **<fieldset>**: Groups related elements in a form.
- **<legend>**: Provides a caption for the <fieldset>.
- **<label>**: Defines a label for an <input> element.
- **<input>**: Defines an input control.
- **<select>**: Defines a drop-down list.
- **<option>**: Defines options in a <select> list.
- **<textarea>**: Defines a multi-line text input control.
- **<datalist>**: Contains a list of predefined options for an <input>.

- **<pre>**: Displays preformatted text.
- **<br>**: Inserts a line break.
- **<input type="submit">**: Defines a submit button.
- **<input type="reset">**: Defines a reset button.
- **<input type="button">**: Defines a general button.

## CSS Properties

- **form:**
  - **max-width: 800px;** Sets the maximum width of the form to 800 pixels.
  - **margin: auto;** Centers the form horizontally.
  - **padding: 20px;** Adds padding of 20 pixels inside the form.
  - **border: 1px solid #ccc;** Adds a 1-pixel solid border with color #ccc (light gray).
  - **border-radius: 10px;** Rounds the corners of the form with a radius of 10 pixels.
  - **background-color: #f9f9f9;** Sets the background color of the form to #f9f9f9 (light gray).
- **label:**
  - **font-weight: bold;** Sets the font weight of labels to bold.
  - **margin-right: 10px;** Adds a right margin of 10 pixels to labels.
- **input[type="text"], input[type="email"], input[type="password"], input[type="number"], input[type="date"], input[type="file"], input[type="color"], textarea, select:**
  - **width: 100%;** Sets the width of these input fields and elements to 100% of their parent container.
  - **padding: 8px;** Adds padding of 8 pixels inside the input fields and elements.

- margin: 5px 0;; Adds vertical margins of 5 pixels.
- box-sizing: border-box;; Includes padding and border in the element's total width and height.
- **textarea:**
  - resize: vertical;; Allows the textarea to be resized vertically only.

## Code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="../Lect3/Lect2.css" />
    <style>
      body {
        background-color: beige;
      }
    </style>
    <title>Lect2</title>
  </head>
  <body>
    <div id="FormGivenInClass">
      <h1 id="FormTitle">Practice Form with all the Fields</h1>
      <form action="">
        <fieldset>
          <legend>Personal Info:</legend>
          <input
            type="hidden"
            name="dbUUID"
            value="aslkfaslkdfjas;lkfjasl;fjl;sajfs"
          />
          <label for="name">Name: </label>
          <input type="text" id="name" name="name" required /><br /><br />
          <label for="email">Email: </label>
          <input type="email" id="email" name="email" required /><br /><br />
          <label for="password">Password: </label>
          <input
            type="password"
            id="password"
            name="password"
            required
          /><br /><br />
          <label for="contact">Contact: </label>
          <input type="number" name="contact" id="contact" /><br /><br />
          <label for="address">Address: </label>
          <input type="text" name="address" id="address" /><br /><br />
        </fieldset>
      </form>
    </div>
  </body>
</html>
```

```

<label for="gender">Gender: </label>
<input type="radio" name="gender" id="male-radio" value="Male" />
<label for="male-radio">Male</label>
<input type="radio" name="gender" id="female-radio" value="Female" />
<label for="female-radio">Female</label><br /><br />
<label for="birthdate">Birth Date: </label>
<input type="date" name="birthdate" id="birthdate" /><br /><br />
<label for="profile-pic">Profile Picture: </label>
<input type="file" name="profile-pic" id="profile-pic" /><br /><br />
<label for="fav-color">Favorite Color: </label>
<input type="color" name="fav-color" id="fav-color" /><br /><br />

<!-- input image -->
<label for="fav-image">Favorite Image: </label>
<input
  type="image"
  name="fav-image"
  id="fav-image"
  onclick="window.location.href='../Lect1/20190503_160252.jpg'"
  onclick="window.location.href='../Lect1/20190503_160252.jpg'"
  src="/360_F_27450287_V4ZC1gE3LxftBSYZfyB6h3yixtA4RtDm.jpg"
  style="height: 50px"
  alt="Google Logo"
/><br /><br />
<pre>
<label for="languages">Languages: </label> <input type="checkbox" name="languages"
id="hindi" value="hindi"> <label for="hindi">Hindi</label>
  <input type="checkbox" name="languages" id="gujarati" value="gujarati">
<label for="gujarati">Gujarati</label>
  <input type="checkbox" name="languages" id="english" value="english">
<label for="english">English</label>
  <input type="checkbox" name="languages" id="spanish" value="spanish">
<label for="spanish">Spanish</label>
</pre>
<label for="linkedin-url">LinkedIn URL: </label>
<input type="url" name="linkedin-url" id="linkedin-url" />
</fieldset>

<fieldset>
<legend>Qualifications</legend>
<label for="qualification">Highest Education: </label>
<select name="qualification" id="qualification">
  <option value="10th">10th</option>
  <option value="12th">12th</option>
  <option value="Graduate">Graduate</option>
  <option value="Post-Graduate">Post-Graduate</option>
  <option value="Doctrate">Doctrate</option>
  <option value="Post-Doctrate">Post-Doctrate</option>
</select>
<br /><br />
<label for="marks">Marks: </label>

```

```

<select name="marks" id="marks">
  <option value="percentage">Percentage</option>
  <option value="percentile">Percentile</option>
  <option value="cgpa-10">CGPA Out of 10</option>
  <option value="cgpa-5">CGPA Out of 5</option>
  <option value="cgpa-4">CGPA Out of 4</option>
</select>
<input type="number" name="marks" id="marks-value" /><br /><br />
<label for="resume">Resume: </label>
<input type="file" name="resume" id="resume" /><br /><br />
<label for="skills">Skills: </label>
<textarea name="skills" id="skills" cols="30" rows="3"></textarea>
<br /><br />
<label for="certificate">Certifications: </label>
<textarea
  name="certificate"
  id="certificate-desc"
  cols="30"
  rows="3"
  placeholder="Enter your certifications seprated by comma"
></textarea>
<input type="file" name="certificate" id="certificate" multiple />
<br /><br />
<label for="proficiency-for-job-role"
  >Proficiency in the job role you are applying:
</label>
<input
  type="range"
  name="proficiency-for-job-role"
  id="proficiency-for-job-role"
  min="0"
  max="10"
  step="1"
/>
</fieldset>

<fieldset>
  <legend>Experience</legend>
  <label for="experience">Experience Years: </label>
  <input
    list="experience"
    id="experience-year"
    name="experience-year"
  />
  <datalist id="experience">
    <option value="No Experience">No Experience</option>
    <option value="1-2 Years">1-2 Years</option>
    <option value="2-5 Years">2-5 Years</option>
    <option value="5-10 Years">5-10 Years</option>
    <option value="10+ Years">10+ Years</option></datalist>
  <br /><br />

```

```
<label for="company">Roles and Exprience: </label>
<textarea
  name="company"
  id="company"
  placeholder="Data Analyst - Google - Jan 2015 - Present, ..."
  rows="5"
  cols="50"
></textarea>
<br /><br />
<label for="projects">Projects: </label>
<textarea
  name="projects"
  id="projects"
  placeholder="Project 1 - Description 1, Project 2 - Description 2, ..."
  rows="5"
  cols="50"
></textarea>
<br /><br />
</fieldset>

<fieldset>
  <legend>Interview Date</legend>
  <label for="interview-date">Interview Date: </label>
  <input
    type="datetime-local"
    name="interview-date"
    id="interview-date"
  /><br /><br />
  <label for="interview-time">Optional Time (On Same Day): </label>
  <input
    type="time"
    name="interview-time"
    id="interview-time"
  /><br /><br />
  <label for="feedback">Feedback or Expectations</label>
  <textarea name="feedback" id="feedback" cols="30" rows="3"></textarea>
  <br /><br />
  <input type="submit" value="Apply" />
  <input type="reset" value="Reset" />
  <input type="button" value="Doubt" />
</fieldset>
</form>
</div>
</body>
</html>
```



## External CSS File Content:

```
form {
    max-width: 800px;
    margin: auto;
    padding: 20px;
    border: 1px solid #ccc;
    border-radius: 10px;
    background-color: #f9f9f9;
}

label {
    font-weight: bold;
    margin-right: 10px;
}

input[type="text"], input[type="email"], input[type="password"],
input[type="number"], input[type="date"], input[type="file"], input[type="color"],
textarea, select {
    width: 100%;
    padding: 8px;
    margin: 5px 0;
    box-sizing: border-box;
}

input[type='color'] {
    height: 50px;
}

textarea {
    resize: vertical;
}
```

## Output:

### Practice Form with all the Fields

Personal Info:

Name:

Email:

Password:

Contact:

Address:

Gender:

☐ Male ☐ Female

Birth Date:

dd/mm/yyyy

Profile Picture:

Choose file

No file chosen

Favorite Color:

Favorite Image:

FIND OUT +

Languages:

☐ Hindi  
☐ Gujarati  
☐ English  
☐ Spanish

LinkedIn URL:

Qualifications

Highest Education:

10th

Marks:

Percentage

Resume:

Choose file

No file chosen

Skills:

Certifications:

Enter your certifications seprated by comma

Choose files

No file chosen

Proficiency in the job role you are applying:

Experience

Experience Years:

Roles and Exprience:

Data Analyst - Google - Jan 2015 - Present, ...

Projects:

Project 1 - Description 1, Project 2 - Description 2, ...

Interview Date

**Interview Date:**

**Optional Time (On Same Day):**

**Feedback or Expectations**

Apply

Reset

Doubt

## Practical-3

Q1. Interactive HTML, CSS and JS webpage.

A1.

### First Part

#### HTML Tags Used

- **<!DOCTYPE html>**: Defines the document type and version of HTML.
- **<html>**: The root element of the HTML document.
- **<head>**: Contains meta-information about the document, including title and style.
- **<meta>**: Provides metadata such as character encoding and viewport settings.
  - **charset="UTF-8"**: Sets the character encoding to UTF-8.
  - **name="viewport"**: Configures the viewport for responsive design.
- **<title>**: Sets the title of the document, which appears in the browser tab.
- **<style>**: Contains internal CSS styles that apply to the document.
- **<body>**: Contains the main content of the document.
- **<div>**: Defines a section or division in the document.
  - **id="coordinates"**: An identifier used to reference the element in JavaScript and CSS.
- **<script>**: Contains JavaScript code that runs in the document.
  - Contains an event listener that updates the coordinates displayed in the **<div>** when the mouse is moved.

#### JavaScript Code Explanation:

```
const coordinatesDiv = document.getElementById('coordinates');
```

- **const coordinatesDiv**: Declares a constant variable named coordinatesDiv.

- **document.getElementById('coordinates')**: Selects the HTML element with the id of "coordinates" from the DOM (Document Object Model). This function returns a reference to the element, which is then assigned to the coordinatesDiv variable. The const keyword ensures that coordinatesDiv cannot be reassigned to a different value later.

```
document.addEventListener('mousemove', function(event) {
```

- **document.addEventListener**: Adds an event listener to the document object. This method listens for events and executes the provided function when the event occurs.
- **'mousemove'**: Specifies the type of event to listen for, in this case, when the mouse is moved.
- **function(event) {...}**: Defines an anonymous function (a function without a name) that will be executed every time the specified event (mousemove) occurs. The event parameter represents the event object that contains details about the event.

```
const x = event.clientX;
```

- **const x**: Declares a constant variable named x.
- **event.clientX**: Retrieves the horizontal coordinate (x) of the mouse pointer relative to the viewport (the visible area of the web page). This value is assigned to the x variable. clientX provides the position of the mouse cursor in pixels from the left edge of the viewport.

```
const y = event.clientY;
```

- **const y**: Declares a constant variable named y.
- **event.clientY**: Retrieves the vertical coordinate (y) of the mouse pointer relative to the viewport. This value is assigned to the y variable. clientY provides the position of the mouse cursor in pixels from the top edge of the viewport.

```
coordinatesDiv.innerHTML = `X: ${x}, Y: ${y}`;
```

- **coordinatesDiv.innerHTML**: Accesses the innerText property of the coordinatesDiv element. This property represents the text content inside the element.

- ``X: ${x}, Y: ${y}``: Uses a template literal (enclosed in backticks ```) to create a string that includes dynamic values. `${x}` and `${y}` are placeholders that are replaced with the values of the `x` and `y` variables, respectively. The resulting string is in the format `"X: [x], Y: [y]"`.
- `=`: Assigns the generated string to the `innerText` property of `coordinatesDiv`, thereby updating the displayed text to show the current mouse coordinates.

### Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Mouse Coordinates</title>
  <style>
    body {
      margin: 0;
      padding: 0;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      font-family: Arial, sans-serif;
      background-color: #f0f0f0;
    }

    #coordinates {
      font-size: 24px;
      background-color: #fff;
      padding: 20px;
      border: 2px solid #000;
      border-radius: 10px;
    }
  </style>
</head>
<body>

  <div id="coordinates">X: 0, Y: 0</div>

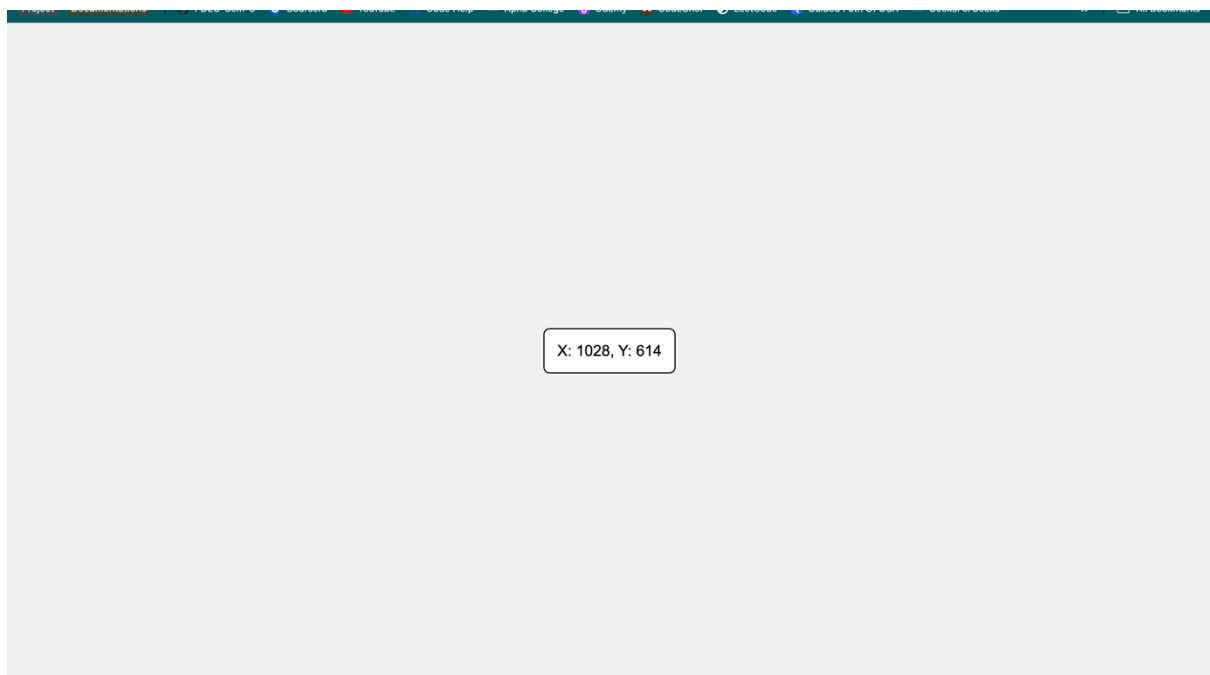
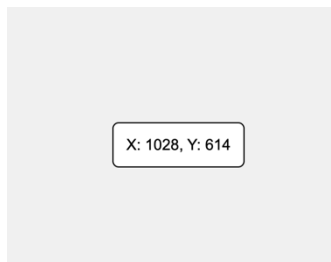
  <script>
    const coordinatesDiv = document.getElementById('coordinates');

    document.addEventListener('mousemove', function(event) {
      const x = event.clientX;
      const y = event.clientY;
      coordinatesDiv.innerText = `X: ${x}, Y: ${y}`;
    });
```

```
</script>

</body>
</html>
```

## Output:



## Part Two

### HTML Tags and Their Descriptions

#### 1. `<!DOCTYPE html>`

- **Description:** Declaration that defines the document type and version of HTML (HTML5 in this case).

#### 2. `<html lang="en">`

- **Description:** Root element of the HTML document.
- **Attributes:**
  - `lang="en"`: Specifies the language of the document as English.

#### 3. `<head>`

- **Description:** Contains meta-information about the HTML document, such as its character set, viewport settings, title, and linked stylesheets or scripts.

#### 4. `<meta charset="UTF-8">`

- **Description:** Specifies the character encoding for the document.
- **Attributes:**
  - `charset="UTF-8"`: Sets the character encoding to UTF-8.

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

- **Description:** Controls the layout on mobile browsers.
- **Attributes:**
  - `name="viewport"`: Specifies the viewport settings.
  - `content="width=device-width, initial-scale=1.0"`: Sets the width of the viewport to the device's width and the initial zoom level to 1.

#### 6. `<title>`

- **Description:** Sets the title of the HTML document, which appears in the browser's title bar or tab.
- **Content:** Java Script Methods and Operators



7. **<link rel="stylesheet" href="JSMethodsAndOps.css">**

- **Description:** Links to an external CSS stylesheet to style the document.
- **Attributes:**
  - rel="stylesheet": Specifies the relationship between the current document and the linked document.
  - href="JSMethodsAndOps.css": URL of the CSS file.

8. **<body>**

- **Description:** Contains the content of the HTML document, such as headings, paragraphs, images, and other elements.

9. **<h1>**

- **Description:** Defines the main heading of the document.
- **Content:** String Methods and Operators

10. **<div class="inputFor">**

- **Description:** A container element with a class attribute used to group related elements together.
- **Attributes:**
  - class="inputFor": Specifies the class of the div for styling.

11. **<div class="tempInput">**

- **Description:** A nested container element with a class attribute.
- **Attributes:**
  - class="tempInput": Specifies the class for styling.

12. **<input type="text" id="StringA" placeholder="Enter String A" value="hello">**

- **Description:** Defines a single-line text input field.
- **Attributes:**
  - type="text": Specifies the input type.

- `id="StringA"`: Provides a unique identifier for the input field.
- `placeholder="Enter String A"`: Shows placeholder text in the input field.
- `value="hello"`: Sets the default value of the input field.

13. **`<button id="strMetOps" onclick="displayResults()">String Method and Operations</button>`**

- **Description:** Defines a clickable button.
- **Attributes:**
  - `id="strMetOps"`: Provides a unique identifier for the button.
  - `onclick="displayResults()"`: Specifies a JavaScript function to run when the button is clicked.
- **Content:** String Method and Operations

14. **`<div id="outputForStr">`**

- **Description:** A container element for displaying the output related to string methods and operations.
- **Attributes:**
  - `id="outputForStr"`: Provides a unique identifier for the div.

15. **`<table>`**

- **Description:** Defines a table.

16. **`<thead>`**

- **Description:** Groups the header content in a table.

17. **`<th>Description</th>`**

- **Description:** Defines a header cell in a table.
- **Content:** Description

18. **`<th>Method/Operation</th>`**

- **Description:** Defines a header cell in a table.

- **Content:** Method/Operation

19. **<th>Result</th>**

- **Description:** Defines a header cell in a table.
- **Content:** Result

20. **<tbody id="strTable">**

- **Description:** Groups the body content in a table.
- **Attributes:**
  - **id="strTable":** Provides a unique identifier for the tbody element.

21. **<br>**

- **Description:** Inserts a line break.

22. **<input type="text" id="NumberA" placeholder="Enter Number A" value="9">**

- **Description:** Defines a single-line text input field for entering numbers.
- **Attributes:**
  - **type="text":** Specifies the input type.
  - **id="NumberA":** Provides a unique identifier for the input field.
  - **placeholder="Enter Number A":** Shows placeholder text in the input field.
  - **value="9":** Sets the default value of the input field.

23. **<button id="strMetOps" onclick="displayResultsForNum()">Number Method and Operations</button>**

- **Description:** Defines a clickable button.
- **Attributes:**
  - **id="strMetOps":** Provides a unique identifier for the button.
  - **onclick="displayResultsForNum()":** Specifies a JavaScript function to run when the button is clicked.

- **Content:** Number Method and Operations

24. **<div id="outputForNum">**

- **Description:** A container element for displaying the output related to number methods and operations.
- **Attributes:**
  - `id="outputForNum"`: Provides a unique identifier for the div.

25. **<input type="text" id="ArrayA" placeholder="Enter Array A (comma-separated)" value="1,2,3,4,5">**

- **Description:** Defines a single-line text input field for entering array elements.
- **Attributes:**
  - `type="text"`: Specifies the input type.
  - `id="ArrayA"`: Provides a unique identifier for the input field.
  - `placeholder="Enter Array A (comma-separated)"`: Shows placeholder text in the input field.
  - `value="1,2,3,4,5"`: Sets the default value of the input field.

26. **<button id="arrMetOps" onclick="displayResultsForArray()">Array Method and Operations</button>**

- **Description:** Defines a clickable button.
- **Attributes:**
  - `id="arrMetOps"`: Provides a unique identifier for the button.
  - `onclick="displayResultsForArray()"`: Specifies a JavaScript function to run when the button is clicked.
- **Content:** Array Method and Operations

27. **<div id="outputForArray">**

- **Description:** A container element for displaying the output related to array methods and operations.

- **Attributes:**

- `id="outputForArray"`: Provides a unique identifier for the div.

28. `<script src="JSMethodsAndOps.js"></script>`

- **Description:** Links to an external JavaScript file.

- **Attributes:**

- `src="JSMethodsAndOps.js"`: URL of the JavaScript file.

## CSS Properties and Their Explanations

1. `body { background-color: beige; }`

- **Description:** Sets the background color of the entire page to beige.

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

- **Description:** Centers the text of all `<h1>` elements within the page.

3. `.inputFor { display: flex; flex-direction: column; align-items: center; gap: 30px; }`

- **Description:** Styles elements with the class `inputFor` to use Flexbox layout.

- **Attributes:**

- `display: flex;`: Uses Flexbox layout for child elements.
- `flex-direction: column;`: Arranges child elements in a column.
- `align-items: center;`: Centers child elements horizontally within the container.
- `gap: 30px;`: Adds a 30-pixel gap between child elements.

4. `.tempInput { display: flex; gap: 40px; }`

- **Description:** Styles elements with the class `tempInput` to use Flexbox layout.

- **Attributes:**

- `display: flex;`: Uses Flexbox layout for child elements.
- `gap: 40px;`: Adds a 40-pixel gap between child elements.

5. **input { border: 1px solid black; height: 30px; border-radius: 10px; text-align: center; }**

- **Description:** Styles all <input> elements.
- **Attributes:**
  - **border: 1px solid black;** Adds a 1-pixel solid black border around the input fields.
  - **height: 30px;** Sets the height of the input fields to 30 pixels.
  - **border-radius: 10px;** Rounds the corners of the input fields with a 10-pixel radius.
  - **text-align: center;** Centers the text within the input fields.

6. **button { border-radius: 10px; height: 40px; font-size: 0.80rem; }**

- **Description:** Styles all <button> elements.
- **Attributes:**
  - **border-radius: 10px;** Rounds the corners of the button with a 10-pixel radius.
  - **height: 40px;** Sets the height of the button to 40 pixels.
  - **font-size: 0.80rem;** Sets the font size of the button text to 0.80 rem (relative to the root element's font size).

7. **table { margin: 0 auto; border-collapse: collapse; width: 50%; }**

- **Description:** Styles all <table> elements.
- **Attributes:**
  - **margin: 0 auto;** Centers the table horizontally within its container by setting top and bottom margins to 0 and left and right margins to auto.
  - **border-collapse: collapse;** Collapses table borders into a single border.
  - **width: 50%;** Sets the table width to 50% of its containing element.

8. **th { border: 1px solid black; padding: 10px; background-color: #f2f277; }**

- **Description:** Styles all <th> elements (table headers).
- **Attributes:**
  - border: 1px solid black;; Adds a 1-pixel solid black border around table headers.
  - padding: 10px;; Adds 10 pixels of padding inside table headers.
  - background-color: #f2f277;; Sets the background color of table headers to a light yellow (#f2f277).

9. **td { border: 1px solid black; text-align: center; padding: 10px; }**

- **Description:** Styles all <td> elements (table data cells).
- **Attributes:**
  - border: 1px solid black;; Adds a 1-pixel solid black border around table cells.
  - text-align: center;; Centers the text within table cells.
  - padding: 10px;; Adds 10 pixels of padding inside table cells.

10. **h3 { text-align: center; }**

- **Description:** Centers the text of all <h3> elements within the page.

11. **#outputForStr { display: none; }**

- **Description:** Hides the element with the ID outputForStr.
- **Attributes:**
  - display: none;; Prevents the element from being displayed on the page.

12. **#outputForNum { display: none; }**

- **Description:** Hides the element with the ID outputForNum.
- **Attributes:**

- `display: none;`: Prevents the element from being displayed on the page.

### 13. `#outputForArray { display: none; }`

- **Description:** Hides the element with the ID `outputForArray`.
- **Attributes:**
  - `display: none;`: Prevents the element from being displayed on the page.

## JavaScript Explanation

### 1. Defining `resultArray`

```
var resultArray = [document.getElementById("outputForStr"),  
document.getElementById("outputForNum")];
```

- **Description:** This creates an array called `resultArray` containing two elements: the HTML elements with the IDs `outputForStr` and `outputForNum`. These elements are referenced so they can be manipulated later.

### 2. Logging `resultArray`

```
console.log(resultArray);
```

- **Description:** Logs the `resultArray` to the console. This helps in debugging by showing the referenced elements.

### 3. `displayResults` Function

```
function displayResults() {  
  
  resultArray.forEach((element) => {  
  
    element.style.display = 'block';  
  
  });  
}
```



```
const strA = document.getElementById("StringA").value;

const strB = document.getElementById("StringB").value;

document.getElementById('outputForStr').style.display = 'block';


const strMethAndAns = [

    // Array of objects containing string methods and their results

];


const tableBody = document.getElementById("strTable");

strMethAndAns.forEach((item) => {

    const row = document.createElement("tr");

    if (tableBody.rows.length % 2 === 0) {

        row.style.backgroundColor = "#f5f5dc";

    } else {

        row.style.backgroundColor = "#f5f5af";

    }

    const descCell = document.createElement("td");

    descCell.textContent = item.desc;

    row.appendChild(descCell);

    const methodCell = document.createElement("td");
```

```

methodCell.textContent = item.method;

row.appendChild(methodCell);

const outputCell = document.createElement("td");

outputCell.textContent = Array.isArray(item.output)

    ? JSON.stringify(item.output)

    : item.output;

row.appendChild(outputCell);

strTable.appendChild(row);

});

}

```

- **Description:**

- **Display Results Elements:** Makes sure that the elements referenced in resultArray are visible by setting their display style to block.
- **Retrieve Input Values:** Gets the values from input fields with IDs StringA and StringB.
- **Define String Methods and Results:** Creates an array strMethAndAns with objects containing descriptions, method calls, and results for various string operations.
- **Populate Table:** Iterates over strMethAndAns to create and append rows to a table with ID strTable. Each row contains three cells: description, method, and output. Rows have alternating background colors for readability.

#### 4. **displayResultsForNum Function**

```
function displayResultsForNum() {
```

```
resultArray.forEach((element) => {  
  
    element.style.display = "block";  
  
});
```

```
const numA = parseFloat(document.getElementById("NumberA").value);  
  
const numB = parseFloat(document.getElementById("NumberB").value);  
  
document.getElementById("outputForNum").style.display = "block";
```

```
const numMethAndAns = [  
  
    // Array of objects containing numeric methods and their results  
  
];
```

```
const numTableBody = document.getElementById("numTable");  
  
numMethAndAns.forEach((item) => {  
  
    const row = document.createElement("tr");  
  
    if (numTableBody.rows.length % 2 === 0) {  
  
        row.style.backgroundColor = "#f5f5dc";  
  
    } else {  
  
        row.style.backgroundColor = "#f5f5af";  
  
    }  
  
}
```

```

const descCell = document.createElement("td");

descCell.textContent = item.desc;

row.appendChild(descCell);

const methodCell = document.createElement("td");

methodCell.textContent = item.method;

row.appendChild(methodCell);

const outputCell = document.createElement("td");

outputCell.textContent = Array.isArray(item.output)

    ? JSON.stringify(item.output)

    : item.output;

row.appendChild(outputCell);

numTableBody.appendChild(row);

});

}

```

- **Description:**

- **Display Results Elements:** Similar to the displayResults function, it makes sure the elements in resultArray are visible.
- **Retrieve Numeric Input Values:** Gets numeric values from input fields with IDs NumberA and NumberB, converting them to floats.
- **Define Numeric Methods and Results:** Creates an array numMethAndAns with objects containing descriptions, method calls, and results for various numeric operations.

- **Populate Numeric Table:** Iterates over numMethAndAns to create and append rows to a table with ID numTable. Each row contains three cells: description, method, and output. Rows have alternating background colors for readability.

## 5. displayResultsForArray Function

```
function displayResultsForArray() {  
  
  resultArray.forEach((element) => {  
  
    element.style.display = "block";  
  
  });  
  
  
  const arrA = document.getElementById("ArrayA").value.split(",").map(Number);  
  
  const arrB = document.getElementById("ArrayB").value.split(",").map(Number);  
  
  document.getElementById("outputForArray").style.display = "block";  
  
  
  const arrMethAndAns = [  
  
    // Array of objects containing array methods and their results  
  
  ];  
  
  
  const arrayTableBody = document.getElementById("arrayTable");  
  
  arrMethAndAns.forEach((item) => {  
  
    const row = document.createElement("tr");  
  
    if (arrayTableBody.rows.length % 2 === 0) {
```

```

        row.style.backgroundColor = "#f5f5dc";

    } else {

        row.style.backgroundColor = "#f5f5af";

    }

    const descCell = document.createElement("td");

    descCell.textContent = item.desc;

    row.appendChild(descCell);

    const methodCell = document.createElement("td");

    methodCell.textContent = item.method;

    row.appendChild(methodCell);

    const outputCell = document.createElement("td");

    outputCell.textContent = Array.isArray(item.output)

        ? JSON.stringify(item.output)

        : item.output;

    row.appendChild(outputCell);

    arrayTableBody.appendChild(row);

});

}

```

- **Description:**

- **Display Results Elements:** Similar to the previous functions, it ensures that elements in resultArray are visible.

- **Retrieve Array Input Values:** Gets values from input fields with IDs ArrayA and ArrayB, splits the values by commas, and converts them into arrays of numbers.
- **Define Array Methods and Results:** Creates an array arrMethAndAns with objects containing descriptions, method calls, and results for various array operations.
- **Populate Array Table:** Iterates over arrMethAndAns to create and append rows to a table with ID arrayTable. Each row contains three cells: description, method, and output. Rows have alternating background colors for readability.

Sorry

**Output:**

## String Methods and Operators

String Method and Operations

Number Method and Operations

Array Method and Operations

String Methods and Operators Output

Description	Method/Operation	Result
Concatenation Operator	strA + strB	helloworld
String Length Method	strA.length	5
String to Upper Case Method	strA.toUpperCase()	HELLO
String to Lower Case Method	strA.toLowerCase()	hello
String Slice Method	strA.slice(1, 3)	el
String Substring Method	strA.substring(1, 3)	el
String Replace Method	strA.replace('H', 'J')	hello
String Char At Method	strA.charAt(1)	e
String Char Code At Method	strA.charCodeAt(1)	101
String Index Of Method	strA.indexOf('e')	1
String Last Index Of Method	strA.lastIndexOf('l')	3
String Search Method	strA.search('ll')	2
String Match Method	strA.match(/l/g)	["l","l"]
String Split Method	strA.split("")	["h","e","l","l","o"]
String Trim Method	strA.trim()	hello
String Value Of Method	String(strA)	hello
String Locale Compare Method	strA.localeCompare(strB)	-1
String Repeat Method	strA.repeat(2)	hellohello
String Includes Method	strA.includes('ell')	true
String Starts With Method	strA.startsWith('He')	false
String Ends With Method	strA.endsWith('lo')	true



9

11

Number Method and Operations

**Number Methods and Operators Output**

Description	Method/Operation	Result
Addition	numA + numB	20
Subtraction	numA - numB	-2
Multiplication	numA * numB	99
Division	numA / numB	0.8181818181818182
Modulus	numA % numB	9
Exponentiation	Math.pow(numA, numB)	31381059609
Square Root of numA	Math.sqrt(numA)	3
Round numA	Math.round(numA)	9
Ceiling of numA	Math.ceil(numA)	9
Floor of numA	Math.floor(numA)	9
Absolute value of numA	Math.abs(numA)	9
Max of numA and numB	Math.max(numA, numB)	11
Min of numA and numB	Math.min(numA, numB)	9

1,2,3,4,5

6,7,8,9,10

Array Method and Operations

**Array Methods and Operators Output**

Description	Method/Operation	Result
Array Concatenation	arrA.concat(arrB)	[1,2,3,4,5,6,7,8,9,10]

Array Methods and Operators Output

Description	Method/Operation	Result
Array Concatenation	arrA.concat(arrB)	[1,2,3,4,5,6,7,8,9,10]
Array Length	arrA.length	5
Array Push	arrA.push(6)	[1,2,3,4,5,6]
Array Pop	arrA.pop()	[1,2,3,4]
Array Shift	arrA.shift()	[2,3,4,5]
Array Unshift	arrA.unshift(0)	[0,1,2,3,4,5]
Array Slice	arrA.slice(1, 3)	[2,3]
Array Splice	arrA.splice(2, 1)	[1,2,4,5]
Array Join	arrA.join('-')	1-2-3-4-5
Array Reverse	arrA.reverse()	[5,4,3,2,1]
Array Sort	arrA.sort()	[1,2,3,4,5]
Array Includes	arrA.includes(2)	true
Array Index Of	arrA.indexOf(2)	1

## Practical-4

Q1. JS Form Validation.

A1.

### HTML Tags Used:

- **<!DOCTYPE html>**: Declares the document type and version of HTML.
- **<html lang="en" dir="ltr">**: Root element of the HTML document; lang specifies the language, and dir specifies the text direction.
- **<head>**: Contains meta-information about the HTML document.
- **<meta charset="UTF-8">**: Sets the character encoding for the document to UTF-8.
- **<meta name="viewport" content="width=device-width, initial-scale=1.0">**: Ensures the page is responsive and properly scaled on different devices.
- **<title>**: Specifies the title of the HTML document, which appears in the browser tab.
- **<link rel="stylesheet" href="/Lect6.css">**: Links an external CSS file for styling the document.
- **<body>**: Contains the content of the HTML document.
- **<div class="wrapper">**: A container for the registration form, styled with CSS.
- **<h2>**: Defines a level 2 heading; used here as the title of the form.
- **<form id="form" action="#">**: Defines an HTML form; id is used for JavaScript interactions and action specifies where to send the form data.
- **<div class="input-box">**: Container for input fields; styled with CSS.
- **<input type="text" placeholder="Enter your name" id="name">**: Text input field for entering the name; id used for JavaScript.
- **<input type="password" placeholder="Create password" id="passwd">**: Password input field for entering the password.
- **<input type="password" placeholder="Confirm password" id="confPasswd">**: Password input field for confirming the password.
- **<div class="policy">**: Container for the terms and conditions checkbox; styled with CSS.

- **<input type="checkbox" id="terms">**: Checkbox input for accepting terms and conditions.
- **<h3>**: Defines a level 3 heading; used here for the terms and conditions text.
- **<div class="input-box button">**: Container for the submit button; styled with CSS.
- **<input type="submit" value="Register Now" id="submitBut">**: Submit button for the form; id used for JavaScript.
- **<script>**: Contains JavaScript code for form validation and interaction.

### CSS Properties:

- **Universal Selector (\*)**:
  - **margin: 0;** - Resets the default margin for all elements to zero.
  - **padding: 0;** - Resets the default padding for all elements to zero.
  - **box-sizing: border-box;** - Includes padding and border in the element's total width and height.
  - **font-family: sans-serif;** - Applies a sans-serif font to all elements.
- **body**:
  - **min-height: 100vh;** - Ensures the body takes up at least the full height of the viewport.
  - **display: flex;** - Uses Flexbox to layout child elements.
  - **align-items: center;** - Centers child elements vertically.
  - **justify-content: center;** - Centers child elements horizontally.
  - **background: #e5e6e7;** - Sets a light gray background color for the body.
- **.wrapper**:
  - **max-width: 430px;** - Sets the maximum width of the wrapper to 430 pixels.
  - **width: 100%;** - Ensures the wrapper takes up the full width of its container.
  - **background: #fff;** - Sets a white background color for the wrapper.
  - **padding: 34px;** - Adds 34 pixels of padding inside the wrapper.

- **border-radius: 6px;** - Rounds the corners of the wrapper with a 6-pixel radius.
- **box-shadow: 0 5px 10px rgba(0,0,0,0.2);** - Adds a subtle shadow around the wrapper for depth.
- **.wrapper h2:**
  - **font-size: 22px;** - Sets the font size of the h2 element to 22 pixels.
  - **font-weight: 600;** - Applies a semi-bold font weight to the h2 element.
  - **color: #333;** - Sets the text color of the h2 element to a dark gray.
- **.wrapper form:**
  - **margin-top: 30px;** - Adds a 30-pixel margin above the form.
- **.wrapper form .input-box:**
  - **height: 52px;** - Sets the height of the input boxes to 52 pixels.
  - **margin: 18px 0;** - Adds a vertical margin of 18 pixels around the input boxes.
- **form .input-box input:**
  - **height: 100%;** - Makes the input field fill the height of its container.
  - **width: 100%;** - Makes the input field fill the width of its container.
  - **outline: none;** - Removes the default outline when the input field is focused.
  - **padding: 0 15px;** - Adds 15 pixels of padding on the left and right sides of the input field.
  - **font-size: 17px;** - Sets the font size of the input text to 17 pixels.
  - **font-weight: 400;** - Applies a normal font weight to the input text.
  - **color: #333;** - Sets the text color of the input field to a dark gray.
  - **border: 1px solid #dedede;** - Adds a 1-pixel solid border with a light gray color.

- **border-radius: 6px;** - Rounds the corners of the input field with a 6-pixel radius.
  - **transition: all 0.3s ease;** - Adds a smooth transition effect for changes in all properties over 0.3 seconds.
- **.input-box input:focus:**
  - **border-color: #4070f4;** - Changes the border color to a blue shade when the input field is focused.
- **.input-box input:valid:**
  - **border-color: #1aad00;** - Changes the border color to green when the input field contains valid data.
- **form .policy:**
  - **display: flex;** - Uses Flexbox to layout the child elements of the policy section.
- **form h3:**
  - **color: #707070;** - Sets the text color of h3 elements to a medium gray.
  - **font-size: 14px;** - Sets the font size of h3 elements to 14 pixels.
  - **font-weight: 500;** - Applies a medium font weight to h3 elements.
  - **margin-left: 10px;** - Adds a 10-pixel left margin to h3 elements.
- **.input-box.button input:**
  - **color: #fff;** - Sets the text color of the button to white.
  - **letter-spacing: 1px;** - Adds 1 pixel of spacing between letters.
  - **border: none;** - Removes the border of the button.
  - **background: #35c828;** - Sets the background color of the button to a green shade.
  - **cursor: pointer;** - Changes the cursor to a pointer when hovering over the button.
- **.input-box.button input:hover:**

- **background: #2fb123;** - Changes the background color of the button to a darker green shade on hover.
- **form .text h3:**
  - **color: #333;** - Sets the text color of h3 elements within .text to a dark gray.
  - **width: 100%;** - Makes the h3 element take up the full width of its container.
  - **text-align: center;** - Centers the text within the h3 element.

### JS Functionalities:

**const validate = (e) => { ... }:**

Defines a function named validate that takes an event object e as its parameter. This function is used to validate the form inputs when the form is submitted.

- **e.preventDefault();**  
This method prevents the default form submission behavior, allowing the function to perform validation checks before the form is actually submitted.

```
const name = document.getElementById('name');
const email = document.getElementById('email');
const password = document.getElementById('passwd');
const confPasswd = document.getElementById('confPasswd');
const terms = document.getElementById('terms');
```

These lines retrieve the HTML elements with the specified IDs and assign them to constants. This allows the JavaScript code to interact with these elements (e.g., check their values).

### // Email validation

- **if (email.value === '') {**  
Checks if the email input field is empty.
  - **alert('Email is required');**  
Displays an alert message if the email field is empty.

- **return;**

Exits the function early if the condition is true, preventing further validation checks.

#### **// Password validation**

- **if (password.value === '') {**

Checks if the password input field is empty.

- **alert('Password is required');**

Displays an alert message if the password field is empty.

- **return;**

Exits the function early if the condition is true.

#### **// Confirm password validation**

- **if (confPasswd.value === '') {**

Checks if the confirm password input field is empty.

- **alert('Confirm Password is required');**

Displays an alert message if the confirm password field is empty.

- **return;**

Exits the function early if the condition is true.

#### **// Check if passwords match**

- **if (password.value !== confPasswd.value) {**

Checks if the password and confirm password fields do not match.

- **alert('Passwords do not match');**

Displays an alert message if the passwords do not match.



- **return;**  
Exits the function early if the condition is true.

#### // Terms acceptance validation

- **if (!terms.checked) {**  
Checks if the terms and conditions checkbox is not checked.
  - **alert('Please accept terms and conditions');**  
Displays an alert message if the checkbox is not checked.
  - **return;**  
Exits the function early if the condition is true.

#### // Regular expressions for validation

- **const emailRE = /^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\$/;**  
Defines a regular expression to validate the email format.
- **const nameRE = /^[A-Za-z\s]+\$/;**  
Defines a regular expression to validate names, allowing only letters and spaces.
- **const passRE = /^(?=.\*[A-Z])(?=.\*\d){9,}\$/;**  
Defines a regular expression to validate passwords, ensuring at least 9 characters, one uppercase letter, and one digit.

#### // Validate email format

- **if (!emailRE.test(email.value)) {**  
Tests if the email input value does not match the email regular expression.
  - **alert('Email is invalid');**  
Displays an alert message if the email format is invalid.
  - **return;**  
Exits the function early if the condition is true.

#### // Validate name format

- **if (!nameRE.test(name.value)) {**

Tests if the name input value does not match the name regular expression.

- **alert('Name is invalid');**

Displays an alert message if the name format is invalid.

- **return;**

Exits the function early if the condition is true.

#### // Validate password format

- **if (!passRE.test(password.value)) {**

Tests if the password input value does not match the password regular expression.

- **alert('Password is invalid. Must be at least 9 characters, contain 1 uppercase letter, and 1 number.');**

Displays an alert message if the password format is invalid.

- **return;**

Exits the function early if the condition is true.

**alert("Form submitted successfully!");**

Displays an alert message indicating that the form has been successfully submitted if all validation checks pass.

**document.getElementById('submitBut').addEventListener('click', validate);**

Adds an event listener to the submit button with the ID submitBut. When the button is clicked, the validate function is called to perform the form validation.

## Code:

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registration or Sign Up form in HTML CSS</title>
    <link rel="stylesheet" href="./Lect6.css">
  </head>
  <body>
    <div class="wrapper">
      <h2>Registration</h2>
      <form id="form" action="#">
        <div class="input-box">
          <input type="text" placeholder="Enter your name" id="name">
        </div>
        <div class="input-box">
          <input type="text" placeholder="Enter your email" id="email">
        </div>
        <div class="input-box">
          <input type="password" placeholder="Create password" id="passwd">
        </div>
        <div class="input-box">
          <input type="password" placeholder="Confirm password" id="confPasswd">
        </div>
        <div class="policy">
          <input type="checkbox" id="terms">
          <h3>I accept all terms & conditions</h3>
        </div>
        <div class="input-box button">
          <input type="submit" value="Register Now" id="submitBut">
        </div>
      </form>
    </div>

    <script>
      const validate = (e) => {
        e.preventDefault(); // Prevent form submission

        const name = document.getElementById('name');
        const email = document.getElementById('email');
        const password = document.getElementById('passwd');
        const confPasswd = document.getElementById('confPasswd');
        const terms = document.getElementById('terms');

        // Email validation
        if (email.value === '') {
          alert('Email is required');
          return;
        }
      }
    </script>
  </body>
</html>
```

```

// Password validation
if (password.value === '') {
    alert('Password is required');
    return;
}

// Confirm password validation
if (confPasswd.value === '') {
    alert('Confirm Password is required');
    return;
}

// Check if passwords match
if (password.value !== confPasswd.value) {
    alert('Passwords do not match');
    return;
}

// Terms acceptance validation
if (!terms.checked) {
    alert('Please accept terms and conditions');
    return;
}

// Regular expressions for validation
const emailRE = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;
const nameRE = /^[A-Za-z\s]+$/; // Names can contain spaces and letters
only
const passRE = /^(?=.*[A-Z])(?=.*\d){9,}$/; // At least 9 characters, 1
uppercase, 1 digit

// Validate email format
if (!emailRE.test(email.value)) {
    alert('Email is invalid');
    return;
}

// Validate name format
if (!nameRE.test(name.value)) {
    alert('Name is invalid');
    return;
}

// Validate password format
if (!passRE.test(password.value)) {
    alert('Password is invalid. Must be at least 9 characters, contain 1
uppercase letter, and 1 number.');
```

```
        alert("Form submitted successfully!");
    };

    document.getElementById('submitBut').addEventListener('click', validate);
</script>
</body>
</html>
```

### External CSS File:

```
*{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: sans-serif;
}
```

```
body{
    min-height: 100vh;
    display: flex;
    align-items: center;
    justify-content: center;
    background: #e5e6e7;
}
```

```
.wrapper{
    max-width: 430px;
    width: 100%;
    background: #fff;
    padding: 34px;
    border-radius: 6px;
    box-shadow: 0 5px 10px rgba(0,0,0,0.2);
}
```

```
.wrapper h2{
    font-size: 22px;
    font-weight: 600;
    color: #333;
}
```

```
.wrapper form{
    margin-top: 30px;
}
```

```
.wrapper form .input-box{
    height: 52px;
    margin: 18px 0;
}
```

```
form .input-box input{
    height: 100%;
    width: 100%;
    outline: none;
```

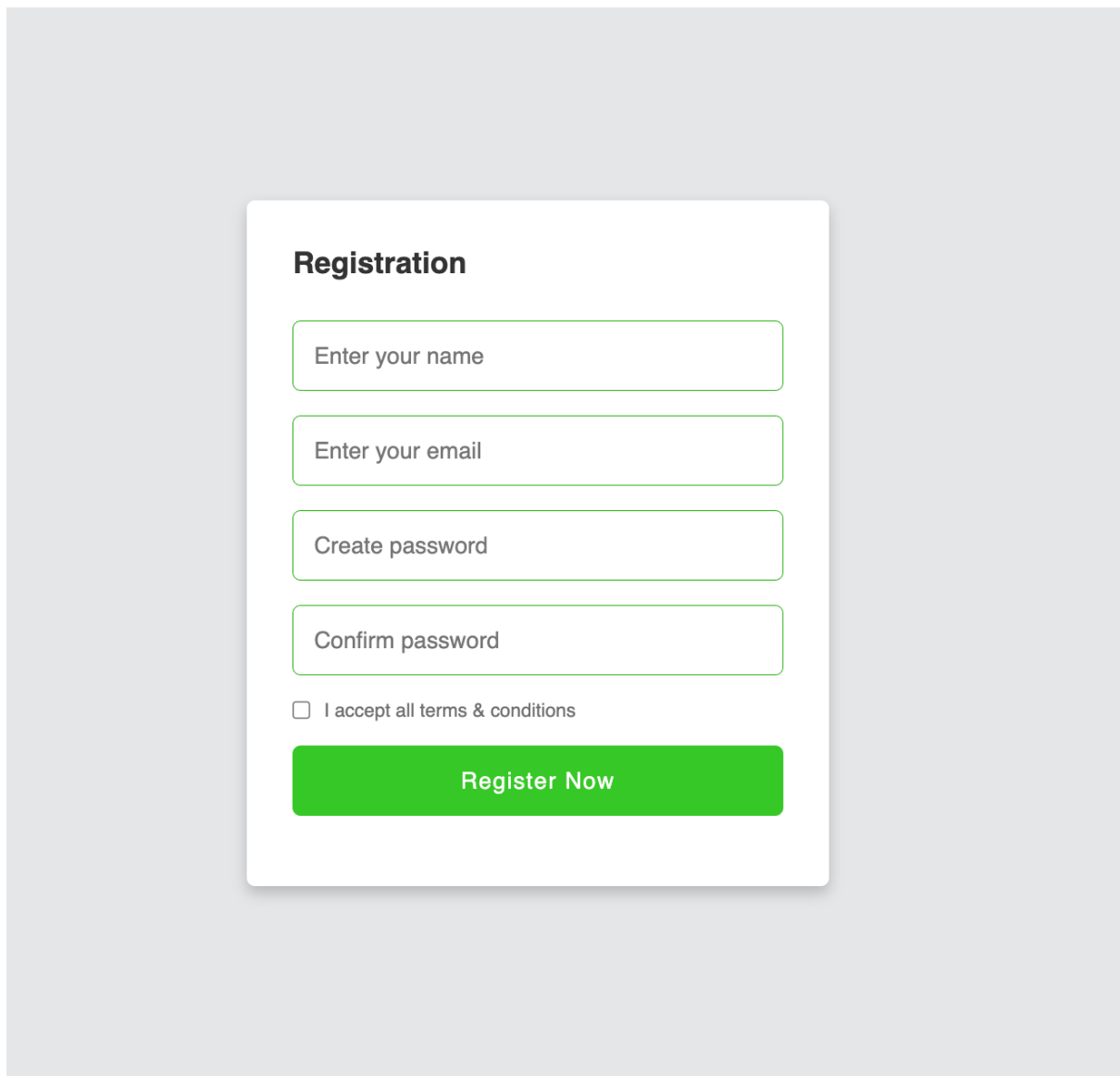
```
padding: 0 15px;
font-size: 17px;
font-weight: 400;
color: #333;
border: 1px solid #dedede;
border-radius: 6px;
transition: all 0.3s ease;
}

.input-box input:focus {
  border-color: #4070f4;
}
.input-box input:valid {
  border-color: #1aad00;
}
form .policy{
  display: flex;
}

form h3{
  color: #707070;
  font-size: 14px;
  font-weight: 500;
  margin-left: 10px;
}

.input-box.button input{
  color: #fff;
  letter-spacing: 1px;
  border: none;
  background: #35c828;
  cursor: pointer;
}
.input-box.button input:hover{
  background: #2fb123;
}
form .text h3{
  color: #333;
  width: 100%;
  text-align: center;
}
```

## Output:

A registration form UI mockup centered on a light gray background. The form is a white card with rounded corners and a subtle drop shadow. It features a title, four input fields with green borders, a checkbox, and a green submit button.

**Registration**

Enter your name

Enter your email

Create password

Confirm password

☐ I accept all terms & conditions

Register Now

127.0.0.1:5500 says

Email is invalid

OK

Registration

Tirth Shah

klbxjlbf

.....

.....

☒ I accept all terms & conditions

Register Now



127.0.0.1:5500 says

Form submitted successfully!

OK

## Registration

Tirth Shah

klbxjlblf@gmail.com

.....

.....

☒ I accept all terms & conditions

Register Now