**JavaScript Assignment - 03**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Exp 3 - String Operations (Revised)</title>

    <style>

        body {

            font-family: 'Segoe UI', Arial, sans-serif;

            background: linear-gradient(93deg, #e0eaef 0%, #f3f2f4 100%);

            margin: 0;

            min-height: 100vh;

        }

        .container {

            max-width: 670px;

            margin: 38px auto;

            background: #fff;

            box-shadow: 0 4px 24px rgba(0,0,0,0.08);

            padding: 30px 32px;

            border-radius: 14px;

        }

        h1 {

            text-align: center;

            color: #1464c4;

            margin-bottom: 10px;

        }

        h2 {

            color: #346078;

            margin-bottom: 8px;

            margin-top: 22px;

        }

        .section {

            background: #f7fbfc;

            border: 1px solid #cde7f6;

            border-radius: 8px;

            margin-bottom: 27px;

            padding: 18px 14px;

        }

        input[type="text"] {

            width: 100%;

            padding: 9px;

            margin-top: 7px;

            border-radius: 5px;

            border: 1px solid #b3c6e0;

            font-size: 1em;

        }

        button {

            background-color: #1464c4;

            color: #fff;

            border: none;

            padding: 9px 18px;

            margin: 10px 6px 0px 0px;

            border-radius: 5px;

            font-weight: 600;

            cursor: pointer;

            transition: background 0.2s;

        }

        button:hover { background: #345479; }

        .reset-btn { background-color: #778899; }

        .reset-btn:hover { background-color: #344050; }

        .output {

            margin-top: 8px;

            font-weight: 500;

            color: #15a735;

            background: #eafbe7;

            border-radius: 4px;

            padding: 6px 10px;

        }

        .error {

            margin-top: 8px;

            font-weight: 500;

            color: #e24863;

            background: #f9dade;

            border-radius: 4px;

            padding: 6px 10px;

        }

        @media(max-width:700px) {

            .container { padding: 9px; }

        }

    </style>

</head>

<body>

<div class="container">

    <h1>JavaScript</h1>

    <div style="text-align:center; margin-bottom:22px;">

        <p>

            <strong>Experiment 3: String operations</strong><br>

            <strong>Soumitra Kode &bullet; Roll : 42134 &bullet; BE-06-                                                                                                                                                                 P6</strong>

        </p>

    </div>

    <!-- Reverse String Section -->

    <div class="section" id="reverseSection">

        <h2>1. Reverse String</h2>

        <input type="text" id="reverseInput" placeholder="Enter a string to reverse">

        <button onclick="reverseString()">Reverse</button>

        <button onclick="resetReverse()" class="reset-btn">Reset</button>

        <div id="reverseOutput" class="output"></div>

        <div id="reverseError" class="error"></div>

    </div>

    <!-- Replace Text Section -->

    <div class="section" id="replaceSection">

        <h2>2. Replace Text</h2>

        <input type="text" id="replaceInput" placeholder="Original string">

        <input type="text" id="replaceFrom" placeholder="Text to find">

        <input type="text" id="replaceTo" placeholder="Replace with">

        <button onclick="replaceCharacters()">Replace All</button>

        <button onclick="resetReplace()" class="reset-btn">Reset</button>

        <div id="replaceOutput" class="output"></div>

        <div id="replaceError" class="error"></div>

    </div>

    <!-- Palindrome Section -->

    <div class="section" id="palindromeSection">

        <h2>3. Check Palindrome</h2>

        <input type="text" id="palindromeInput" placeholder="Enter string to check">

        <button onclick="checkPalindrome()">Check</button>

        <button onclick="resetPalindrome()" class="reset-btn">Reset</button>

        <div id="palindromeOutput" class="output"></div>

        <div id="palindromeError" class="error"></div>

    </div>

</div>

<script>

// Utility function to clear output and error

function clearOutput(sectionId) {

    document.getElementById(sectionId + 'Output').innerText = '';

    document.getElementById(sectionId + 'Error').innerText = '';

}

// 1. Reverse String Function

function reverseString() {

    clearOutput('reverse');

    const input = document.getElementById('reverseInput').value.trim();

    if (!input) {

        document.getElementById('reverseError').innerText = 'Please enter a valid string.';

        return;

    }

    const reversed = input.split('').reverse().join('');

    document.getElementById('reverseOutput').innerText = `Reversed String: ${reversed}`;

}

function resetReverse() {

    document.getElementById('reverseInput').value = '';

    clearOutput('reverse');

}

// 2. Replace Text Function

function replaceCharacters() {

    clearOutput('replace');

    const input = document.getElementById('replaceInput').value;

    const from = document.getElementById('replaceFrom').value;

    const to = document.getElementById('replaceTo').value;

    if (!input.trim()) {

        document.getElementById('replaceError').innerText = 'Original string cannot be empty.';

        return;

    }

    if (!from) {

        document.getElementById('replaceError').innerText = 'Text to find cannot be empty.';

        return;

    }

    const replaced = input.replaceAll(from, to);

    document.getElementById('replaceOutput').innerText = `Result: ${replaced}`;

}

function resetReplace() {

    document.getElementById('replaceInput').value = '';

    document.getElementById('replaceFrom').value = '';

    document.getElementById('replaceTo').value = '';

    clearOutput('replace');

}

// 3. Palindrome Check Function

function checkPalindrome() {

    clearOutput('palindrome');

    const input = document.getElementById('palindromeInput').value.trim();

    if (!input) {

        document.getElementById('palindromeError').innerText = 'Please enter a valid string.';

        return;

    }

    const clean = input.toLowerCase().replace(/[^a-z0-9]/g, '');

    const isPalindrome = clean === clean.split('').reverse().join('');

    document.getElementById('palindromeOutput').innerText = isPalindrome

        ? `"${input}" is a palindrome.`

        : `"${input}" is not a palindrome.`;

}

function resetPalindrome() {

    document.getElementById('palindromeInput').value = '';

    clearOutput('palindrome');

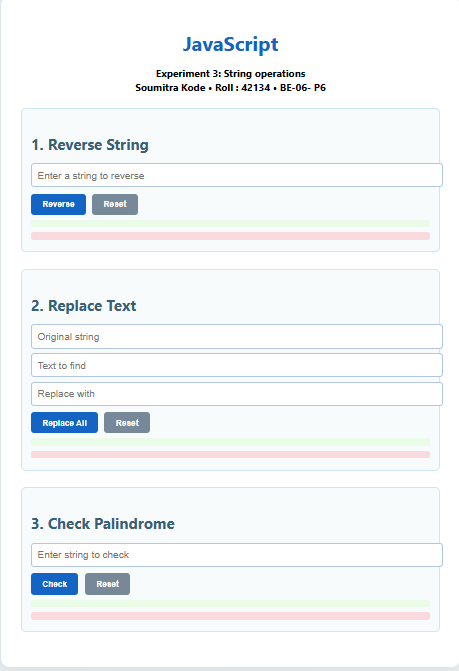
}

</script>

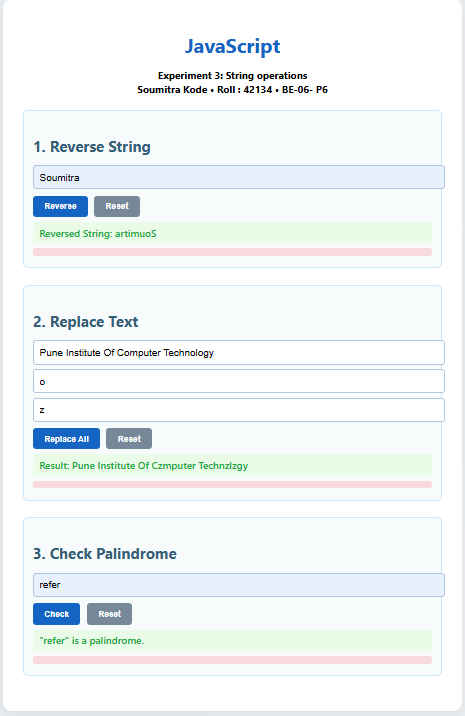
</body>

</html>

Output :   
A. Initial Web Page Layout –



B. Web Page after performing string operations –



**JavaScript Assignment - 04**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Experiment 4 - String Comparison (Modern Style)</title>

  <style>

    body {

      font-family: 'Inter', 'Segoe UI', Arial, sans-serif;

      background: linear-gradient(120deg, #e0eafc 0%, #cfdef3 100%);

      margin: 0;

      min-height: 100vh;

      display: flex;

      align-items: center;

      justify-content: center;

    }

    .container {

      backdrop-filter: blur(8px);

      background: rgba(255, 255, 255, 0.95);

      box-shadow: 0 8px 32px #cbdaeb55, 0 1.5px 7px #b0aad444;

      border-radius: 18px;

      max-width: 420px;

      width: 97vw;

      padding: 38px 30px 28px 30px;

      position: relative;

      animation: floatIn 0.85s cubic-bezier(.91,.03,.33,1.02);

    }

    @keyframes floatIn {

      from { opacity: 0; transform: translateY(30px) scale(.96);}

      to { opacity: 1; transform: none;}

    }

    .student-info {

      font-size: 15px;

      color: #3678bb;

      margin-bottom: 16px;

      background: #f2fcff;

      border-radius: 7px;

      padding: 10px 13px;

      display: flex;

      align-items: center;

      gap: 8px;

    }

    h1 {

      margin: 0 0 7px 0;

      color: #31a4b1;

      letter-spacing: 0.02em;

      font-size: 1.39em;

      font-weight: 700;

    }

    .theory {

      background: #f6f8fc;

      border-left: 5px solid #3ec6c0;

      border-radius: 7px;

      color: #70562e;

      font-size: 1em;

      margin-bottom: 19px;

      padding: 14px 15px 13px 17px;

      line-height: 1.49;

    }

    label {

      display: block;

      margin-top: 17px;

      font-weight: 570;

      color: #1a3658;

      font-size: 1em;

    }

    input[type="text"] {

      width: 100%;

      padding: 10px;

      margin-top: 7px;

      border-radius: 6px;

      border: 2px solid #e2eef7;

      font-size: 1.09em;

      background: #fafdffcc;

      transition: border-color 0.22s;

    }

    input[type="text"]:focus {

      border-color: #42bdd5;

      outline: none;

      background: #e0f5fd;

    }

    .btn-group {

      display: flex;

      flex-wrap: wrap;

      gap: 10px;

      margin-top: 15px;

      margin-bottom: 10px;

      justify-content: center;

    }

    .btn-group button {

      background: linear-gradient(90deg,#3ec6c0 5%,#6db3f2 95%);

      color: #fff;

      padding: 9px 17px;

      border: none;

      border-radius: 6px;

      font-weight: 600;

      cursor: pointer;

      font-size: 1.01em;

      box-shadow: 0 1.5px 7px #bfe8fb44;

      transition: filter 0.16s,transform 0.13s;

      will-change: filter,transform;

    }

    .btn-group button:hover, .btn-group button:focus {

      filter: brightness(1.1) contrast(1.1);

      transform: scale(1.06);

    }

    #result {

      margin-top: 19px;

      padding: 13px 10px 13px 16px;

      background: #f1fefe;

      border-left: 5px solid #3ec6c0;

      min-height: 35px;

      border-radius: 7px;

      color: #22674c;

      font-weight: 650;

      font-size: 1.07em;

      box-shadow: 0 1.2px 5px #edfffa66;

      overflow-x: auto;

      transition: background 0.15s;

    }

    .fade {

      animation: fadeIn 0.7s;

    }

    @keyframes fadeIn {

      from {opacity: 0;}

      to {opacity: 1;}

    }

    @media (max-width: 600px) {

      .container {max-width: 97vw; padding: 13px 4vw;}

    }

  </style>

</head>

<body>

  <div class="container">

    <div class="student-info">

      <span><b>Soumitra Kode</b> | Roll: <b>42134</b> | BE-06 P6</span>

    </div>

    <h1>Experiment 4: String Comparison</h1>

    <div class="theory">

      <strong>String Comparison Types</strong><br>

      <ul style="margin:6px 0 0 15px; padding:0;">

        <li><b>Strict Equality</b>: Returns whether both strings are identical.</li>

        <li><b>Length</b>: States which string (if any) is longer, or if they are equal in length.</li>

        <li><b>Alphabetical Order</b>: Determines dictionary order using locale comparison.</li>

        <li><b>Substring</b>: Indicates if one string appears within the other.</li>

      </ul>

    </div>

    <label for="string1">Enter String 1:</label>

    <input type="text" id="string1" placeholder="Type first string...">

    <label for="string2">Enter String 2:</label>

    <input type="text" id="string2" placeholder="Type second string...">

    <div class="btn-group">

      <button onclick="compareStrings('strict')">Strict Equality</button>

      <button onclick="compareStrings('length')">Length</button>

      <button onclick="compareStrings('locale')">Alphabetical</button>

      <button onclick="compareStrings('substring')">Substring</button>

    </div>

    <div id="result"></div>

  </div>

  <script>

    function compareStrings(method) {

      let str1 = document.getElementById("string1").value;

      let str2 = document.getElementById("string2").value;

      let resultBox = document.getElementById("result");

      let result = "";

      if (method === "strict") {

        result = (str1 === str2)

          ? "Both strings are STRICTLY equal."

          : "Strings are NOT strictly equal.";

      }

      else if (method === "length") {

        if (str1.length > str2.length) {

          result = "String 1 is longer than String 2.";

        } else if (str1.length < str2.length) {

          result = "String 2 is longer than String 1.";

        } else {

          result = "Both strings have equal length.";

        }

      }

      else if (method === "locale") {

        let cmp = str1.localeCompare(str2);

        if (cmp < 0) {

          result = "String 1 comes before String 2 alphabetically.";

        } else if (cmp > 0) {

          result = "String 1 comes after String 2 alphabetically.";

        } else {

          result = "Both strings are alphabetically equal.";

        }

      }

      else if (method === "substring") {

        if (str1 && str2 && str1.includes(str2)) {

          result = "String 2 is a substring of String 1.";

        } else if (str1 && str2 && str2.includes(str1)) {

          result = "String 1 is a substring of String 2.";

        } else {

          result = "Neither string is a substring of the other.";

        }

      }

      resultBox.innerHTML = result;

      resultBox.classList.add('fade');

      setTimeout(() => { resultBox.classList.remove('fade'); }, 700);

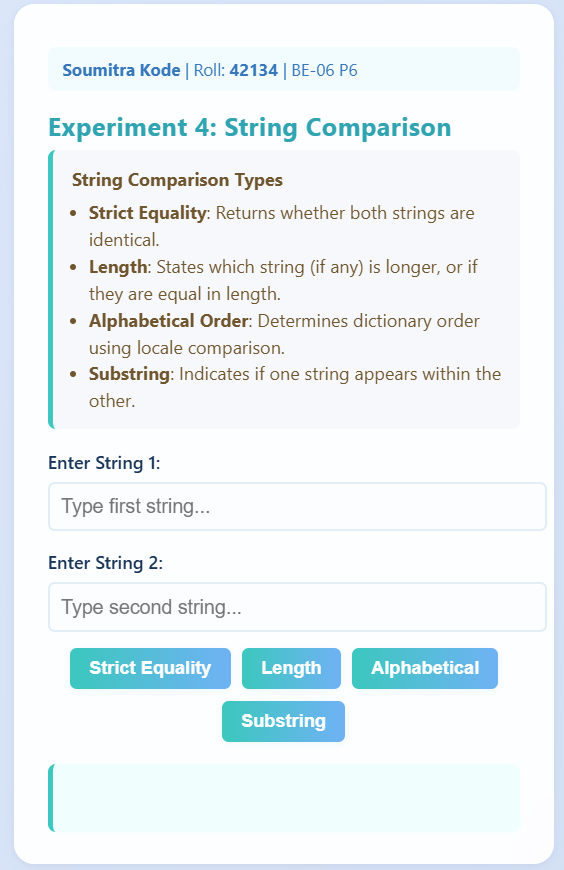
    }

  </script>

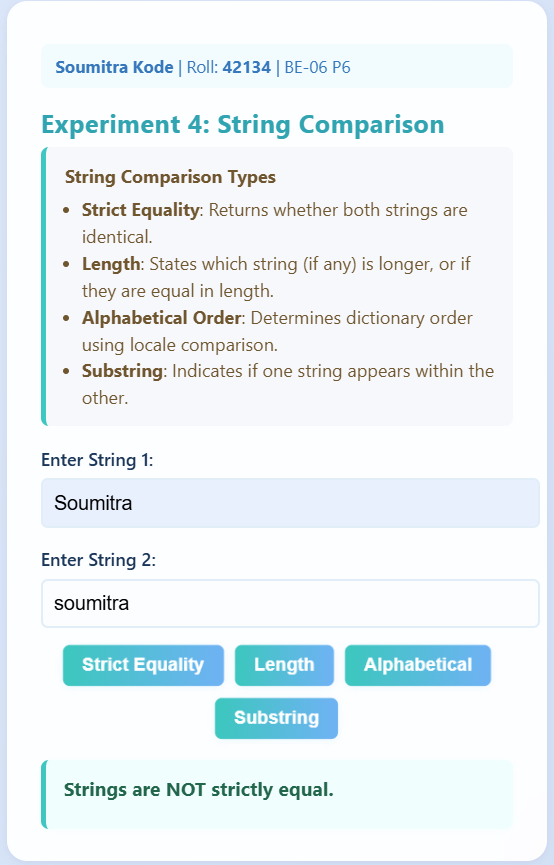
</body>

</html>

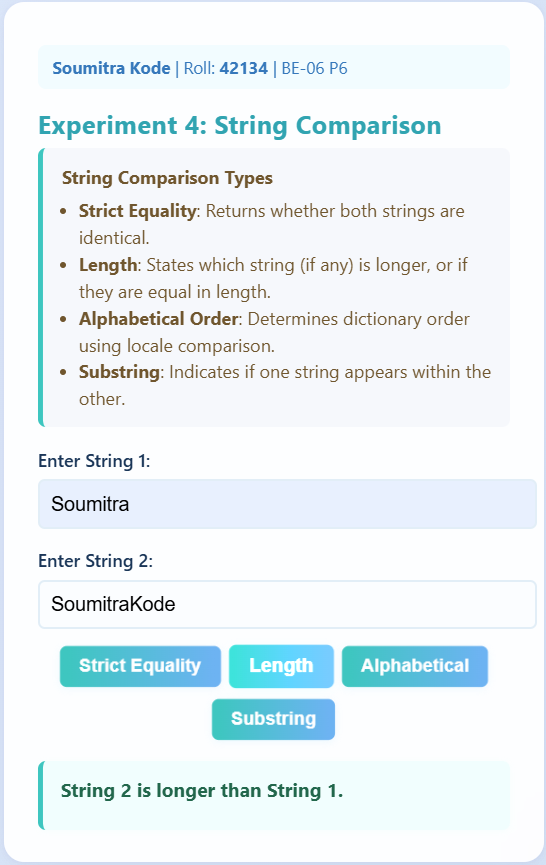
Output :   
A. Initial Web Page Layout –



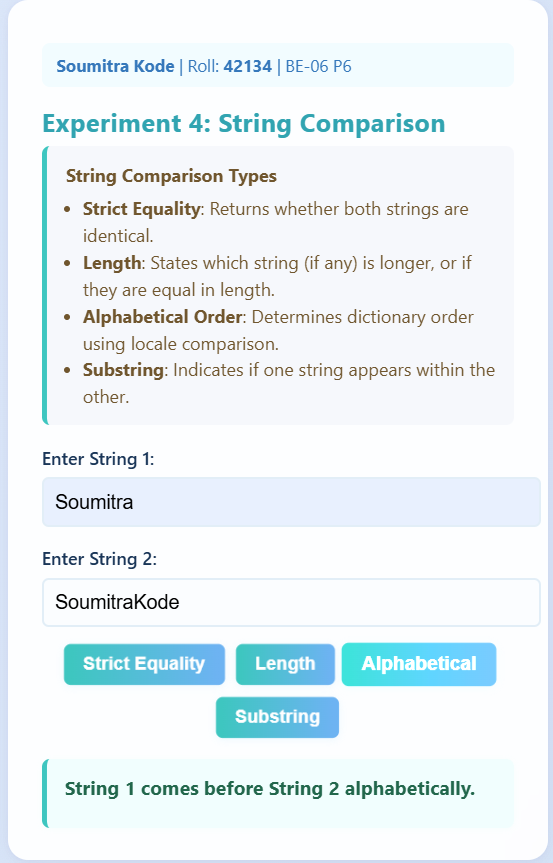
B. Output for strict equal comparison –



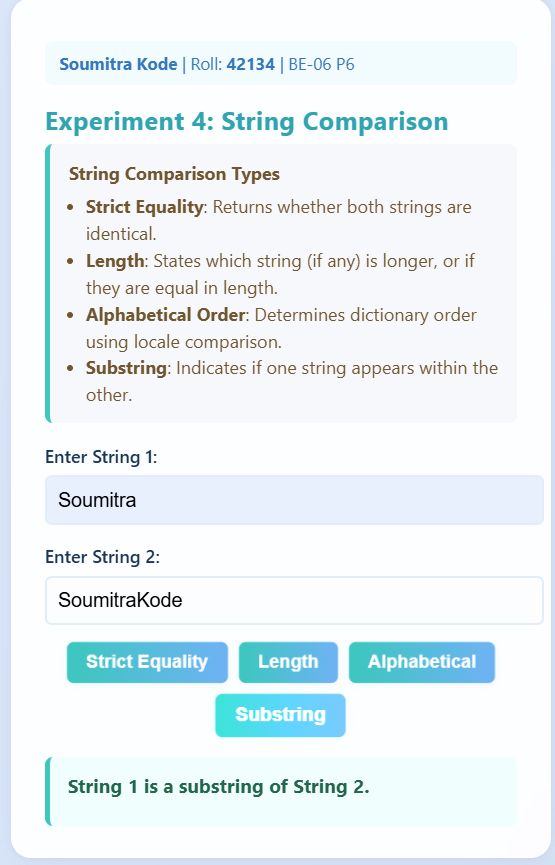
C. Output for comparison of length –



D. Output for Alphabetical precedence –



E. Output for Substring presence –



**JavaScript Assignment - 05**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1" />

  <title>Live Countdown Timer - Modern Style</title>

  <style>

    @import url('https://fonts.googleapis.com/css2?family=Nunito:wght@400;700&display=swap');

    body {

      font-family: 'Nunito', Arial, sans-serif;

      background: linear-gradient(135deg, #d9e2f3 0%, #ffffff 100%);

      margin: 0;

      padding: 40px 20px;

      display: flex;

      justify-content: center;

      min-height: 100vh;

      align-items: center;

    }

    .container {

      background: rgba(255, 255, 255, 0.85);

      border-radius: 18px;

      box-shadow: 0 8px 32px rgba(0,0,0,0.1);

      padding: 40px 38px 32px 38px;

      max-width: 560px;

      width: 95vw;

      text-align: center;

      backdrop-filter: saturate(180%) blur(15px);

      border: 1.5px solid #aac9f7;

      animation: fadeInScale 0.7s ease forwards;

    }

    @keyframes fadeInScale {

      from { opacity: 0; transform: scale(0.92);}

      to { opacity: 1; transform: scale(1);}

    }

    h2 {

      color: #344e86;

      margin-bottom: 18px;

      font-weight: 700;

      font-size: 1.85em;

      letter-spacing: 0.05em;

    }

    .student-info {

      color: #61759f;

      font-size: 1.12em;

      margin-bottom: 24px;

      font-weight: 600;

      letter-spacing: 0.025em;

      line-height: 1.5;

      user-select: none;

    }

    label {

      font-weight: 700;

      color: #1e3a8a;

      display: block;

      font-size: 1.12em;

      margin-bottom: 8px;

      text-align: left;

      max-width: 300px;

      margin-left: auto;

      margin-right: auto;

    }

    input[type="datetime-local"] {

      width: 100%;

      max-width: 300px;

      padding: 14px 16px;

      font-size: 1.1em;

      border: 2px solid #a9bcf5;

      border-radius: 12px;

      outline-offset: 3px;

      outline-color: #668cffcc;

      transition: border-color 0.3s ease, outline-color 0.3s ease;

      margin: 0 auto 24px auto;

      display: block;

      color: #2b3a7f;

      font-weight: 600;

      cursor: pointer;

      user-select: text;

    }

    input[type="datetime-local"]:focus {

      border-color: #3f58b4;

      outline-color: #546ed9;

      background-color: #e6eaff;

    }

    .btn-row {

      display: flex;

      gap: 14px;

      justify-content: center;

      max-width: 340px;

      margin: 0 auto;

    }

    .btn-row button {

      flex: 1;

      padding: 14px 10px;

      background: linear-gradient(135deg, #3b72ff 0%, #246cf3 100%);

      border: none;

      border-radius: 14px;

      color: white;

      font-weight: 700;

      font-size: 1.12em;

      cursor: pointer;

      transition: background-color 0.25s ease, box-shadow 0.3s ease;

      box-shadow: 0 4px 14px rgba(36,108,243,0.45);

      user-select: none;

    }

    .btn-row button:hover, .btn-row button:focus {

      background: linear-gradient(135deg, #5d87ff 0%, #4a78ee 100%);

      box-shadow: 0 6px 23px rgba(36,108,243,0.65);

    }

    #countdown {

      font-size: 3.2em;

      font-weight: 900;

      margin: 36px 0 0 0;

      color: #dc3c45;

      letter-spacing: 0.08em;

      font-family: 'Courier New', Courier, monospace;

      min-height: 64px;

      user-select: none;

    }

    @media (max-width: 400px) {

      h2 { font-size: 1.55em; }

      #countdown { font-size: 2.5em; }

      .btn-row { gap: 10px; }

    }

  </style>

</head>

<body>

  <div class="container">

    <h2>Countdown Timer</h2>

    <div class="student-info">

      Name: <b>Soumitra Kode</b><br>

      Roll No: <b>42134</b><br>

      Div: <b>BE-06</b><br>

      Batch: <b>P6</b>

    </div>

    <label for="endDate">Enter end date and time:</label>

    <input type="datetime-local" id="endDate" />

    <div class="btn-row">

      <button id="startBtn">Start Countdown</button>

      <button id="resetBtn">Reset</button>

    </div>

    <div id="countdown">0d 0h 0m 0s</div>

  </div>

  <script>

    const startBtn = document.getElementById('startBtn');

    const resetBtn = document.getElementById('resetBtn');

    const countdownEl = document.getElementById('countdown');

    const endInput = document.getElementById('endDate');

    let countdownInterval;

    startBtn.addEventListener('click', () => {

      clearInterval(countdownInterval);

      if (!endInput.value) {

        alert('Please select an end date/time.');

        return;

      }

      const endTime = new Date(endInput.value).getTime();

      if (isNaN(endTime)) {

        alert('Invalid date format.');

        return;

      }

      countdownInterval = setInterval(() => {

        const now = new Date().getTime();

        const distance = endTime - now;

        if (distance < 0) {

          clearInterval(countdownInterval);

          countdownEl.textContent = "COUNTDOWN FINISHED!";

          return;

        }

        const days = Math.floor(distance / (1000 \* 60 \* 60 \* 24));

        const hours = Math.floor((distance % (1000 \* 60 \* 60 \* 24)) / (1000 \* 60 \* 60));

        const minutes = Math.floor((distance % (1000 \* 60 \* 60)) / (1000 \* 60));

        const seconds = Math.floor((distance % (1000 \* 60)) / 1000);

        countdownEl.textContent = `${days}d ${hours}h ${minutes}m ${seconds}s`;

      }, 1000);

    });

    resetBtn.addEventListener('click', () => {

      clearInterval(countdownInterval);

      countdownEl.textContent = "0d 0h 0m 0s";

      endInput.value = "";

    });

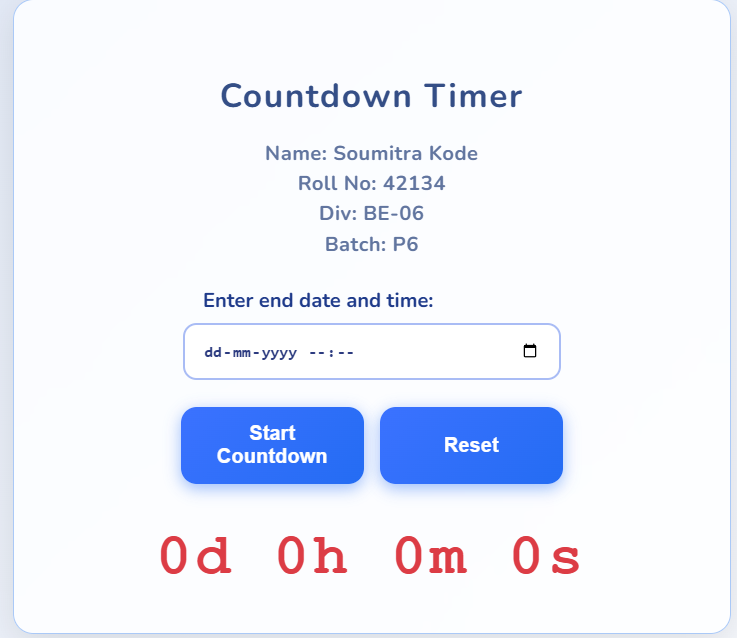
  </script>

</body>

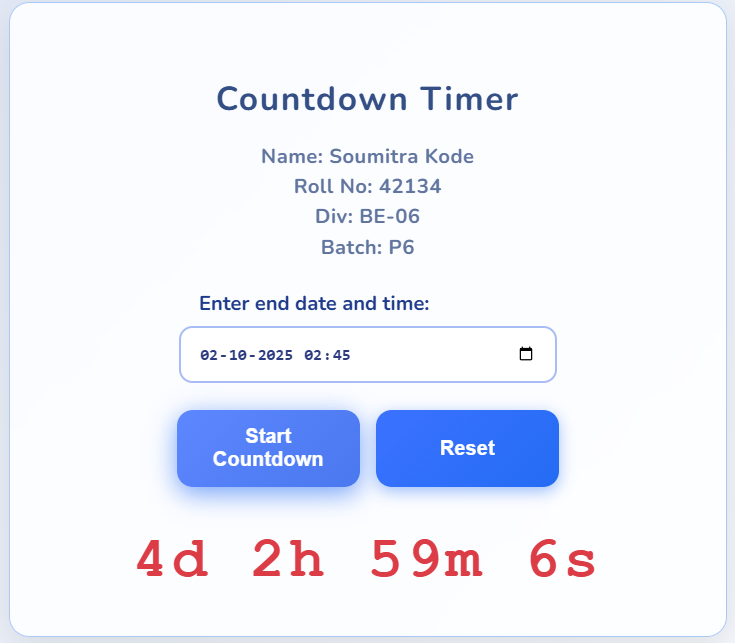
</html>

Output –

1. Initial Landing Page –



1. Set and Start Countdown –



**JavaScript Assignment - 06**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1" />

  <title>Array Operations - Creative Edition</title>

  <style>

    @import url('https://fonts.googleapis.com/css2?family=Work+Sans:wght@400;600&display=swap');

    body {

      background: linear-gradient(135deg, #f0f4f8 0%, #d9e2f3 100%);

      font-family: 'Work Sans', Arial, sans-serif;

      margin: 0;

      padding: 36px 18px;

      display: flex;

      justify-content: center;

      min-height: 100vh;

      align-items: flex-start;

      color: #224166;

      user-select: none;

    }

    .container {

      width: 420px;

      background: #fcfdffcc;

      border-radius: 18px;

      padding: 30px 26px 40px 26px;

      box-shadow: 0 10px 28px #a7bacddd;

      backdrop-filter: saturate(180%) blur(14px);

      border: 1.8px solid #c7d9f3;

      animation: fadeSlideIn 0.7s ease forwards;

    }

    @keyframes fadeSlideIn {

      from {opacity: 0; transform: translateY(30px);}

      to {opacity: 1; transform: translateY(0);}

    }

    h1 {

      font-weight: 700;

      font-size: 1.58em;

      margin-bottom: 14px;

      text-align: center;

      color: #1c2d4a;

      letter-spacing: 0.045em;

      user-select: text;

    }

    h2 {

      font-weight: 650;

      font-size: 1.22em;

      color: #2f466b;

      border-bottom: 2px solid #7ea4d2;

      padding-bottom: 6px;

      margin-bottom: 22px;

      user-select: text;

    }

    label {

      font-weight: 600;

      font-size: 1.02em;

      color: #28384e;

      margin-top: 18px;

      display: inline-block;

      user-select: text;

    }

    input[type="number"] {

      width: 100%;

      padding: 12px 15px;

      font-size: 1.15em;

      border: 2px solid #aac1dc;

      border-radius: 9px;

      margin-top: 6px;

      font-weight: 500;

      transition: border-color 0.3s;

      color: #1e2a47;

    }

    input[type="number"]:focus {

      outline: none;

      border-color: #4575e6;

      background-color: #e7efff;

    }

    button {

      margin-top: 12px;

      font-weight: 700;

      font-size: 1.06em;

      color: #f9faff;

      background: linear-gradient(90deg, #3778f0 10%, #4a63d9 95%);

      border: none;

      border-radius: 12px;

      padding: 11px 16px;

      box-shadow: 0 3px 19px #4a63d947;

      cursor: pointer;

      transition: background 0.24s ease, box-shadow 0.3s ease;

      user-select: none;

      width: 100%;

    }

    button:hover, button:focus {

      background: linear-gradient(90deg, #526bdb 10%, #5366cf 90%);

      box-shadow: 0 6px 22px #525bfd5c;

      outline: none;

      transform: scale(1.03);

    }

    .btn-row {

      display: flex;

      gap: 14px;

      margin-top: 12px;

    }

    .btn-row button {

      width: 50%;

      padding: 14px 0;

      font-weight: 700;

    }

    .section {

      margin-top: 28px;

      user-select: text;

    }

    .array-display {

      background: #e6f0fd;

      border: 1.5px solid #9abcffaa;

      border-radius: 14px;

      padding: 13px 18px;

      font-family: 'Courier New', monospace;

      font-size: 1.12em;

      color: #2350a0;

      min-height: 35px;

      margin-top: 9px;

      overflow-x: auto;

      white-space: nowrap;

      word-wrap: normal;

    }

    #message, #searchMessage {

      margin-top: 20px;

      font-weight: 700;

      font-size: 1.07em;

      color: #2d4f87;

      min-height: 26px;

      user-select:none;

      text-align: center;

      transition: color 0.27s ease;

    }

  </style>

</head>

<body>

  <div class="container" role="main" aria-label="Array Operations Interface">

    <h1>Array Operations</h1>

    <!-- Array Size -->

    <div>

      <label for="arraySize">Enter size of array:</label>

      <input type="number" id="arraySize" min="1" aria-describedby="sizeHelp" aria-label="Array size input">

      <button onclick="setSize()" aria-live="polite">Set Size</button>

    </div>

    <!-- Add Elements -->

    <div id="addElementBox" class="section" style="display:none;">

      <label for="elementInput">Enter element:</label>

      <input type="number" id="elementInput" aria-label="Element input">

      <button onclick="addElement()" aria-live="polite">Add Element</button>

      <p id="sizeMessage" style="margin-top:5px; color:#647a99;" aria-live="polite"></p>

    </div>

    <!-- Current Array Display -->

    <div class="section" aria-live="polite" aria-atomic="true">

      <strong>INPUT ARRAY FOR OPERATIONS:</strong>

      <div id="currentArray" class="array-display">[]</div>

    </div>

    <!-- Remove Element Section -->

    <div class="section">

      <h2>Remove Element</h2>

      <label for="removeValue">Enter value to remove:</label>

      <input type="number" id="removeValue" aria-label="Value to remove">

      <div class="btn-row">

        <button onclick="removeNormal()" aria-live="polite">Remove (Normal method)</button>

        <button onclick="removeSplice()" aria-live="polite">Remove (Using splice)</button>

      </div>

      <strong>Array After Remove Operation:</strong>

      <div id="removeArray" class="array-display">[]</div>

    </div>

    <!-- Search Section -->

    <div class="section">

      <h2>Search for Element</h2>

      <label for="searchValue">Enter value to search:</label>

      <input type="number" id="searchValue" aria-label="Value to search">

      <div class="btn-row">

        <button onclick="searchLinear()" aria-live="polite">Search (Linear Search)</button>

        <button onclick="searchIncludes()" aria-live="polite">Search (Using includes)</button>

      </div>

      <div id="searchMessage" aria-live="polite" style="min-height: 30px; font-weight: 600; color: #395682;"></div>

    </div>

    <!-- Empty Array Section -->

    <div class="section">

      <h2>Empty the Array</h2>

      <div class="btn-row">

        <button onclick="emptyNormal()" aria-live="polite">Empty (Set to [])</button>

        <button onclick="emptySplice()" aria-live="polite">Empty (Using splice)</button>

      </div>

      <strong>Array After Empty Operation:</strong>

      <div id="emptyArray" class="array-display">[]</div>

    </div>

    <p id="message" aria-live="polite" style="margin-top:22px;"></p>

  </div>

  <script>

    let arr = [];

    let maxSize = 0;

    function setSize() {

      maxSize = parseInt(document.getElementById("arraySize").value);

      if (isNaN(maxSize) || maxSize <= 0) {

        showMessage("Please enter a valid array size.");

        return;

      }

      arr = [];

      displayMasterArray();

      clearOperationDisplays();

      clearSearchMessage();

      showMessage(`Array size set to ${maxSize}. Add elements one by one.`);

      document.getElementById("addElementBox").style.display = "block";

      document.getElementById("sizeMessage").textContent = `0 / ${maxSize} elements added.`;

    }

    function addElement() {

      if (arr.length >= maxSize) {

        showMessage("Array is already full.");

        return;

      }

      const val = parseInt(document.getElementById("elementInput").value);

      if (isNaN(val)) {

        showMessage("Please enter a valid integer.");

        return;

      }

      arr.push(val);

      document.getElementById("elementInput").value = "";

      displayMasterArray();

      document.getElementById("sizeMessage").textContent = `${arr.length} / ${maxSize} elements added.`;

      if (arr.length === maxSize) {

        showMessage("Array creation completed.");

      }

    }

    function removeNormal() {

      const val = parseInt(document.getElementById("removeValue").value);

      if (isNaN(val)) {

        showMessage("Please enter a valid number to remove.");

        return;

      }

      const filteredArr = arr.filter(item => item !== val);

      displayMasterArray();

      displayRemoveArray(filteredArr);

      if (filteredArr.length < arr.length) {

        showMessage("Value removed using normal method.");

      } else {

        showMessage("Value not found.");

      }

    }

    function removeSplice() {

      const val = parseInt(document.getElementById("removeValue").value);

      if (isNaN(val)) {

        showMessage("Please enter a valid number to remove.");

        return;

      }

      let newArr = arr.slice();

      const index = newArr.indexOf(val);

      if (index !== -1) {

        newArr.splice(index, 1);

        displayMasterArray();

        displayRemoveArray(newArr);

        showMessage("Value removed using splice.");

      } else {

        displayRemoveArray(newArr);

        showMessage("Value not found.");

      }

    }

    function searchLinear() {

      const val = parseInt(document.getElementById("searchValue").value);

      if (isNaN(val)) {

        showSearchMessage("Please enter a valid number to search.");

        return;

      }

      let foundIndex = -1;

      for (let i = 0; i < arr.length; i++) {

        if (arr[i] === val) {

          foundIndex = i;

          break;

        }

      }

      if (foundIndex !== -1) {

        showSearchMessage(`Element found at index: ${foundIndex}`);

      } else {

        showSearchMessage("Element not found.");

      }

    }

    function searchIncludes() {

      const val = parseInt(document.getElementById("searchValue").value);

      if (isNaN(val)) {

        showSearchMessage("Please enter a valid number to search.");

        return;

      }

      const index = arr.indexOf(val);

      if (index !== -1) {

        showSearchMessage(`Element found at index: ${index}`);

      } else {

        showSearchMessage("Element not found.");

      }

    }

    function emptyNormal() {

      arr = [];

      displayMasterArray();

      displayEmptyArray(arr);

      showMessage("Array emptied using normal method.");

    }

    function emptySplice() {

      arr.splice(0, arr.length);

      displayMasterArray();

      displayEmptyArray(arr);

      showMessage("Array emptied using splice().");

    }

    function displayMasterArray() {

      document.getElementById("currentArray").textContent = JSON.stringify(arr);

    }

    function displayRemoveArray(arrayToShow) {

      document.getElementById("removeArray").textContent = JSON.stringify(arrayToShow);

    }

    function displayEmptyArray(arrayToShow) {

      document.getElementById("emptyArray").textContent = JSON.stringify(arrayToShow);

    }

    function clearOperationDisplays() {

      document.getElementById("removeArray").textContent = "[]";

      document.getElementById("emptyArray").textContent = "[]";

    }

    function showMessage(msg) {

      document.getElementById("message").textContent = msg;

    }

    function showSearchMessage(msg) {

      document.getElementById("searchMessage").textContent = msg;

    }

    function clearSearchMessage() {

      document.getElementById("searchMessage").textContent = "";

    }

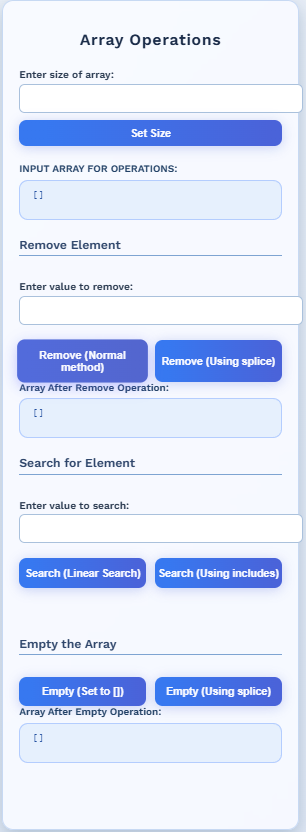
  </script>

</body>

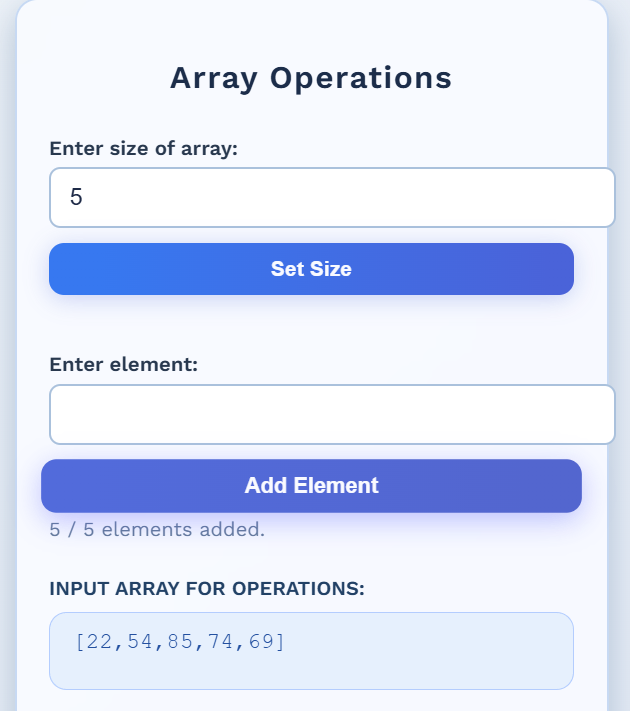
</html>

Output –

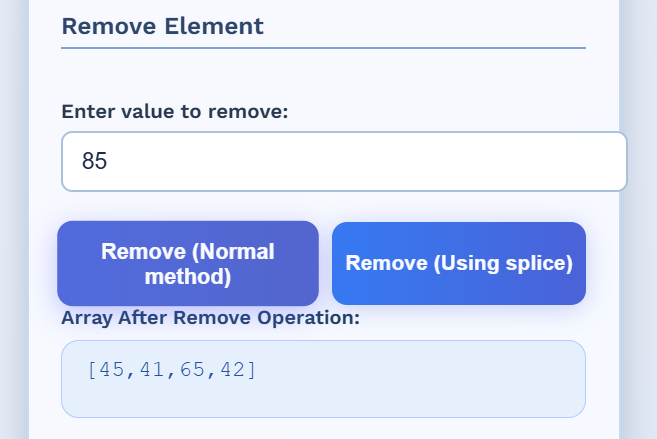
1. Initial Landing Page –



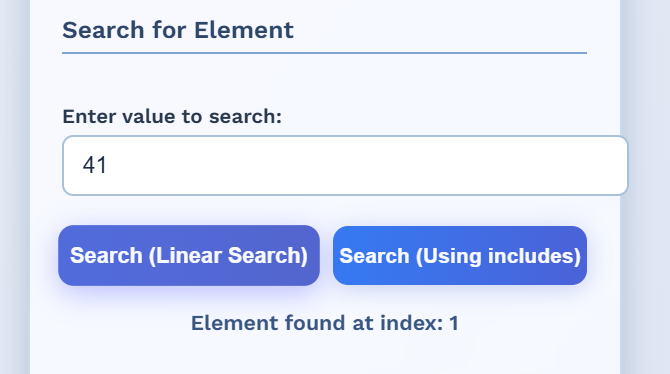
1. Output after initializing array –



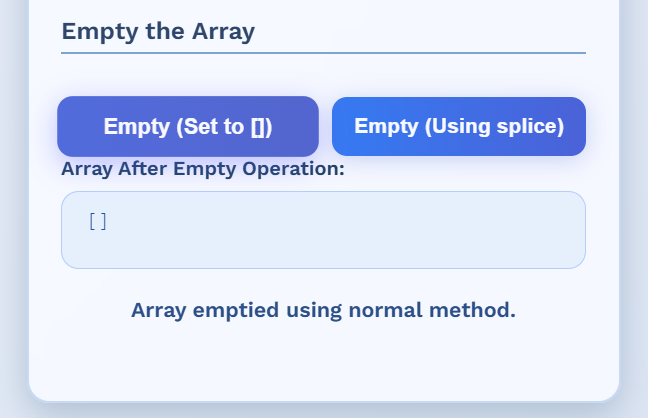
1. Deleting an element from array –



1. Output after searching an element in array –



1. Output after emptying the entire array –



**JavaScript Assignment - 07**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1" />

  <title>Experiment 7: Array Object - Creative Style</title>

  <style>

    @import url('https://fonts.googleapis.com/css2?family=Rubik:wght@400;600&display=swap');

    body {

      margin: 0;

      padding: 20px;

      font-family: 'Rubik', Arial, sans-serif;

      background: linear-gradient(135deg, #f4e8f8, #d5e9f7);

      display: flex;

      justify-content: center;

      min-height: 100vh;

      align-items: center;

      user-select: none;

    }

    .container {

      width: 420px;

      background: rgba(255, 255, 255, 0.82);

      border-radius: 18px;

      padding: 30px 28px 32px 28px;

      box-shadow: 0 12px 36px rgba(148, 128, 187, 0.28);

      text-align: center;

      backdrop-filter: saturate(180%) blur(12px);

      border: 1.8px solid #b5abd4;

      animation: slideFadeIn 0.7s ease forwards;

    }

    @keyframes slideFadeIn {

      from {opacity: 0; transform: translateY(35px);}

      to {opacity: 1; transform: translateY(0);}

    }

    .header {

      background: #b796d8;

      color: #fff;

      border-radius: 9px;

      padding: 21px 14px;

      margin-bottom: 22px;

      font-size: 1.15em;

      font-weight: 600;

      letter-spacing: 0.055em;

      box-shadow: 0 4px 13px #a25fcc99;

      user-select: text;

      line-height: 1.4;

    }

    h2 {

      font-size: 1.4em;

      font-weight: 700;

      margin-bottom: 20px;

      color: #593f8a;

      letter-spacing: 0.025em;

    }

    .row {

      margin-top: 18px;

    }

    input[type="number"],

    input[type="text"] {

      width: 100%;

      padding: 11px 14px;

      border-radius: 12px;

      border: 2px solid #bdb2db;

      font-weight: 500;

      font-size: 1.06em;

      transition: border-color 0.3s, background-color 0.3s;

      color: #483b74;

      font-family: 'Rubik', Arial, sans-serif;

      outline-offset: 2px;

      outline-color: #a496c4cc;

      user-select: text;

    }

    input[type="number"]:focus,

    input[type="text"]:focus {

      border-color: #7e67c0;

      background-color: #eae2ff;

      outline-color: #7461bbaa;

      outline-style: solid;

      outline-width: 1.8px;

    }

    button {

      margin-top: 12px;

      font-weight: 700;

      font-size: 1.07em;

      background: linear-gradient(90deg, #8462e1 10%, #6e4dcc 90%);

      color: #f8f7ff;

      padding: 14px 10px;

      border-radius: 16px;

      border: none;

      box-shadow: 0 4px 24px #6246b670;

      cursor: pointer;

      transition: background-color 0.25s ease, box-shadow 0.35s ease, transform 0.17s ease;

      width: 100%;

      user-select: none;

    }

    button:hover, button:focus {

      background: linear-gradient(90deg, #a07bf4 5%, #805dce 95%);

      box-shadow: 0 6px 29px #715acf92;

      outline: none;

      transform: scale(1.04);

    }

    .hint {

      font-size: 0.84em;

      color: #7869a7;

      margin-top: 6px;

      font-style: italic;

      text-align: left;

      user-select: text;

    }

    .output {

      margin-top: 22px;

      background: #f7f5ff;

      border: 1.5px solid #c4b7e2;

      border-radius: 14px;

      height: 168px;

      overflow-y: auto;

      text-align: left;

      padding: 15px 18px;

      font-family: Consolas, monospace;

      line-height: 1.55;

      color: #5a4d81;

      white-space: pre-wrap;

      user-select: text;

      box-shadow: inset 0 3px 12px #ddd8f6;

    }

  </style>

</head>

<body>

  <main class="container" role="main" aria-label="Array Object Operations Interface">

    <section class="header" aria-label="Student Information">

      <h3>Soumitra Kode</h3>

      <p>Roll No: 42134 | Div: BE-06 | Batch: P6</p>

    </section>

    <h2>Experiment 7: Array Object</h2>

    <section class="row" aria-labelledby="set-size-label">

      <input type="number" id="sizeInput" aria-describedby="size-hint" placeholder="Enter array size (e.g., 6)" />

      <button onclick="setSize()" aria-live="polite" aria-label="Set array size">Set Size</button>

      <div id="size-hint" class="hint">Enter a positive numeric size for the array.</div>

    </section>

    <section class="row" id="addSection" style="display:none;" aria-label="Add Element Section" aria-live="polite">

      <input type="text" id="valueInput" aria-label="Input value to add to array" placeholder='Enter any object value (e.g:[1,2], {"a":10})' />

      <button onclick="appendElement()" aria-label="Add element to array">Add Element</button>

      <div class="hint">

        Use valid JSON for arrays/objects. Keys must be enclosed in double quotes.

      </div>

    </section>

    <section class="row" id="checkSection" style="display:none;" aria-live="polite">

      <input type="number" id="indexInput" aria-label="Enter index to check if element is an array" placeholder="Enter index to check" />

      <button onclick="checkIfArray()" aria-label="Check if element at index is array">Check If Array</button>

    </section>

    <section class="output" id="output" aria-live="polite" aria-atomic="true"></section>

  </main>

  <script>

    let myarr = [];

    let maxSize = 0;

    function setSize() {

      const n = parseInt(document.getElementById("sizeInput").value);

      if (isNaN(n) || n <= 0) {

        showOutput("Please enter a valid positive size!");

        return;

      }

      maxSize = n;

      myarr = [];

      document.getElementById("addSection").style.display = "block";

      document.getElementById("checkSection").style.display = "block";

      showOutput("Size limit = " + maxSize);

      showOutput("Current Array: " + JSON.stringify(myarr));

    }

    function appendElement() {

      if (maxSize === 0) {

        showOutput("Set the array size first!");

        return;

      }

      if (myarr.length >= maxSize) {

        showOutput("Overflow! Cannot add more than " + maxSize + " elements.");

        return;

      }

      const raw = document.getElementById("valueInput").value.trim();

      if (!raw) {

        showOutput("Please enter a value!");

        return;

      }

      let value;

      try {

        value = JSON.parse(raw);

      } catch {

        value = raw;

      }

      myarr.push(value);

      showOutput("Element added: " + JSON.stringify(value));

      showOutput("Current Array (" + myarr.length + "/" + maxSize + "): " + JSON.stringify(myarr));

      document.getElementById("valueInput").value = "";

    }

    function checkIfArray() {

      const idx = parseInt(document.getElementById("indexInput").value);

      if (isNaN(idx) || idx < 0 || idx >= myarr.length) {

        showOutput("Invalid index!");

        return;

      }

      const el = myarr[idx];

      if (Array.isArray(el)) {

        showOutput("Element at index " + idx + " IS an array: " + JSON.stringify(el));

      } else {

        showOutput("Element at index " + idx + " is NOT an array (" + typeof el + "). Value: " + JSON.stringify(el));

      }

    }

    function showOutput(msg) {

      const out = document.getElementById("output");

      out.innerHTML += msg + "\n\n";

      out.scrollTop = out.scrollHeight;

    }

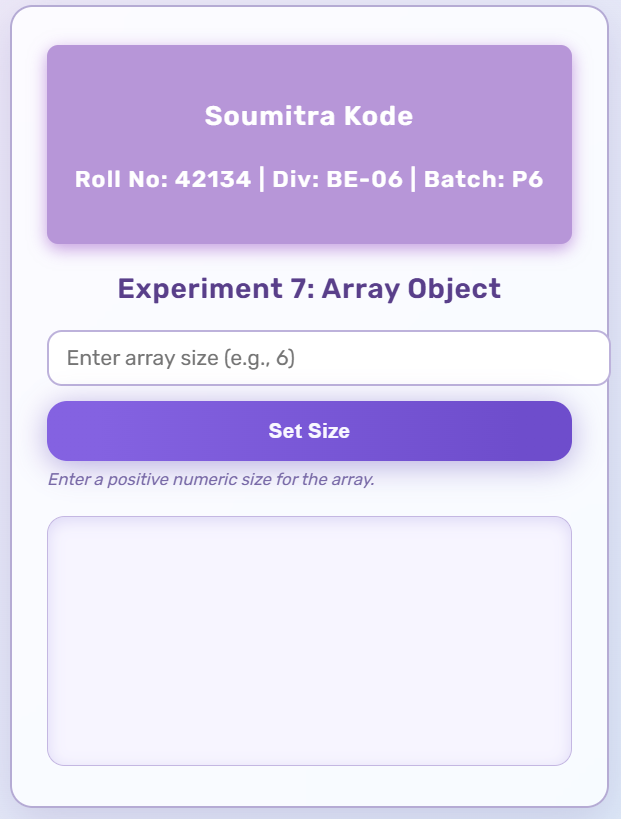
  </script>

</body>

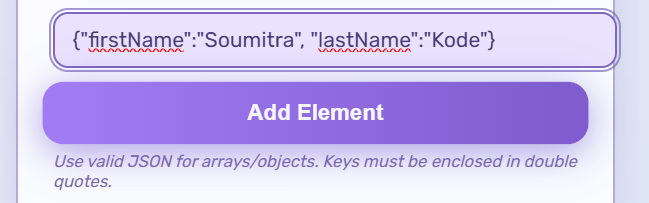
</html>

Output –

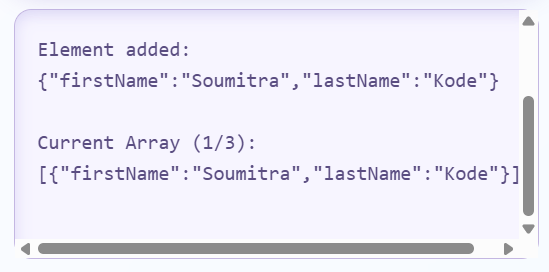
1. Initial Landing page –



1. Adding Element to the array –
2. Input Element.



1. Current array.



1. Getting element at a index –



**JavaScript Assignment - 08**

Name – Soumitra Anil Kode

Roll no. – 42134 , BE- 06, Batch - P6

DOP :

**Git Repo Link :**

**Source Code :**