Program 1

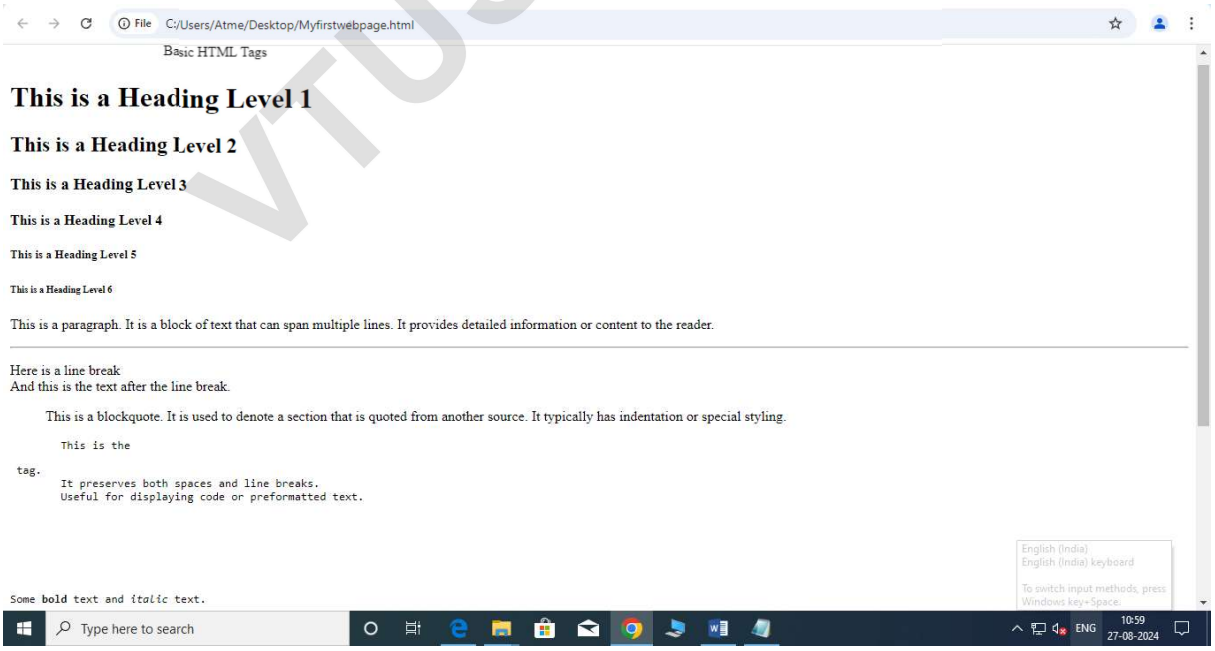
1. Develop the HTML page named as “Myfirstwebpage.html”. Add the following tags with relevant content
   1. Set the title of the page as “My First Web Page”
   2. Within the body use the following tags:
      a) Moving text = “Basic HTML Tags”
      b) Different heading tags (h1 to h6)
      c) Paragraph
      d) Horizontal line
      e) Line Break
      f) Block Quote
      g) Pre tag
      h) Different Logical Style (<b>, <u>, <sub>, <sup> etc.)

Program

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My First Web Page</title>
</head>
<body>
  <!-- Moving text -->
  <marquee>Basic HTML Tags</marquee>

  <!-- Different heading tags -->
  <h1>This is a Heading Level 1</h1>
  <h2>This is a Heading Level 2</h2>
  <h3>This is a Heading Level 3</h3>
  <h4>This is a Heading Level 4</h4>
  <h5>This is a Heading Level 5</h5>
  <h6>This is a Heading Level 6</h6>

  <!-- Paragraph -->
  <p>This is a paragraph. It is a block of text that can span multiple lines. It provides detailed information or content to the reader.</p>

  <!-- Horizontal line -->
  <hr>

  <!-- Line Break -->
  <p>Here is a line break<br>And this is the text after the line break.</p>
```

```
<!-- Block Quote -->
  <blockquote>
    <p>This is a blockquote. It is used to denote a section that is quoted from another source. It typically has indentation or special styling.</p>
  </blockquote>

  <!-- Pre tag -->
  <pre>
    This is the <pre> tag.
    It preserves both spaces and line breaks.
    Useful for displaying code or preformatted text.
  </pre>

  <!-- Different Logical Style -->
  <p>Some <b>bold</b> text and <i>italic</i> text.</p>
  <p>Here is some <u>underlined</u> text.</p>
  <p>Here is a subscript: H<sub>2</sub>O</p>
  <p>Here is a superscript: E = mc<sup>2</sup></p>

</body>
</html>
```

Output



Explanation of the Tags Used:

1. **<title>**: Sets the title of the page that appears on the browser tab.
2. **<marquee>**: Creates scrolling text (Note: <marquee> is obsolete and not recommended for modern web practices, but included here as per the requirement).
3. **<h1> to <h6>**: Represent different levels of headings.
4. **<p>**: Defines a paragraph.
5. **<hr>**: Inserts a horizontal line.
6. **
**: Adds a line break.
7. **<blockquote>**: Used for longer quotations or to indicate a block of quoted text.
8. **<pre>**: Displays preformatted text, preserving both spaces and line breaks.
9. **Logical styles**: <b>, <i>, <u>, <sub>, and <sup> are used for bold, italic, underline, subscript, and superscript text, respectively.

Program 2

2. Develop the HTML page named as "Table.html" to display your class time table.
   a) Provide the title as Time Table with table header and table footer, row-span and col-span etc.
   b) Provide various colour options to the cells (Highlight the lab hours and elective hours with different colours.)
   c) Provide colour options for rows.

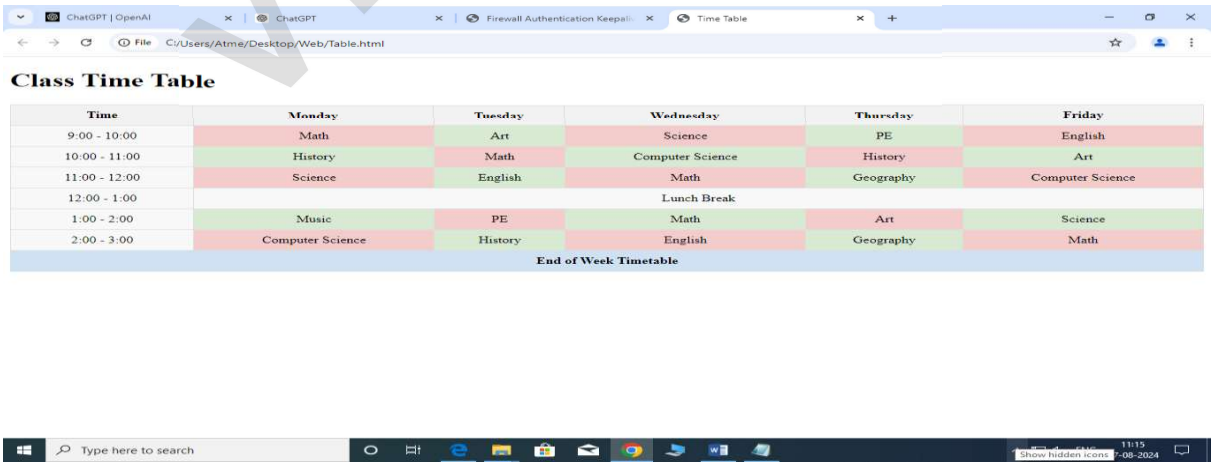
Program

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Time Table</title>
  <style>
    table {
      width: 100%;
      border-collapse: collapse;
    }
    th, td {
      border: 1px solid #ddd;
      padding: 8px;
      text-align: center;
    }
    th {
      background-color: #f2f2f2;
    }
    .lab-hour {
      background-color: #f4cccc; /* Light red for lab hours */
    }
    .elective-hour {
      background-color: #d9ead3; /* Light green for elective hours */
    }
    .highlight-row {
      background-color: #f9f9f9; /* Light grey for rows */
    }
    .header {
      background-color: #b6d7a8; /* Light green for header */
      font-weight: bold;
    }
    .footer {
      background-color: #cfe2f3; /* Light blue for footer */
      font-weight: bold;
    }
  </style>
```

```
</head>
<body>
  <h1>Class Time Table</h1>

  <table>
    <!-- Table Header -->
    <thead>
      <tr class="header">
        <th>Time</th>
        <th>Monday</th>
        <th>Tuesday</th>
        <th>Wednesday</th>
        <th>Thursday</th>
        <th>Friday</th>
      </tr>
    </thead>
    <!-- Table Body -->
    <tbody>
      <tr class="highlight-row">
        <td>9:00 - 10:00</td>
        <td class="lab-hour">Math</td>
        <td class="elective-hour">Art</td>
        <td class="lab-hour">Science</td>
        <td class="elective-hour">PE</td>
        <td class="lab-hour">English</td>
      </tr>
      <tr class="highlight-row">
        <td>10:00 - 11:00</td>
        <td class="elective-hour">History</td>
        <td class="lab-hour">Math</td>
        <td class="elective-hour">Computer Science</td>
        <td class="lab-hour">History</td>
        <td class="elective-hour">Art</td>
      </tr>
      <tr class="highlight-row">
        <td>11:00 - 12:00</td>
        <td class="lab-hour">Science</td>
        <td class="elective-hour">English</td>
        <td class="lab-hour">Math</td>
        <td class="elective-hour">Geography</td>
        <td class="lab-hour">Computer Science</td>
      </tr>
      <tr class="highlight-row">
        <td>12:00 - 1:00</td>
        <td colspan="5">Lunch Break</td>
      </tr>
      <tr class="highlight-row">
        <td>1:00 - 2:00</td>
```

```
            <td class="elective-hour">Music</td>
            <td class="lab-hour">PE</td>
            <td class="elective-hour">Math</td>
            <td class="lab-hour">Art</td>
            <td class="elective-hour">Science</td>
          </tr>
          <tr class="highlight-row">
            <td>2:00 - 3:00</td>
            <td class="lab-hour">Computer Science</td>
            <td class="elective-hour">History</td>
            <td class="lab-hour">English</td>
            <td class="elective-hour">Geography</td>
            <td class="lab-hour">Math</td>
          </tr>
        </tbody>
        <!-- Table Footer -->
        <tfoot>
          <tr class="footer">
            <td colspan="6">End of Week Timetable</td>
          </tr>
        </tfoot>
      </table>

</body>
</html>
```


Output



Explanation of the HTML and CSS:

1. `<title>`: Sets the title of the page as "Time Table."
2. **Table Header (`<thead>`)**:
   - Contains the column titles for the timetable.
3. **Table Body (`<tbody>`)**:
   - Contains the timetable data.
   - Uses `class="highlight-row"` to style alternating rows.
   - Utilizes classes like `lab-hour` and `elective-hour` to apply different background colors for lab and elective hours.
   - Uses `colspan` to merge cells across multiple columns for the lunch break.
4. **Table Footer (`<tfoot>`)**:
   - Provides a footer row that spans all columns with a summary or note about the table.
5. **CSS Styles**:
   - Define the appearance of the table, including borders, padding, and background colors for different types of hours and rows.

Program 3

3. Develop an external style sheet named as "style.css" and provide different styles for h2, h3, hr, p, div, span, time, img & a tags. Apply different CSS selectors for tags and demonstrate the significance of each.

Program

style.css

```
/* Universal selector for basic styling */
* {
   margin: 0;
   padding: 0;
   box-sizing: border-box;
}

/* Styling for h2 tag */
h2 {
   color: #2c3e50; /* Dark blue color */
   font-family: Arial, sans-serif;
   font-size: 24px;
   margin-bottom: 20px;
}

/* Styling for h3 tag */
h3 {
   color: #16a085; /* Teal color */
   font-family: 'Courier New', Courier, monospace;
   font-size: 20px;
   text-transform: uppercase;
}

/* Styling for horizontal rule */
hr {
   border: none;
   height: 2px;
   background-color: #3498db; /* Blue color */
   margin: 20px 0;
}

/* Styling for p tag */
p {
   color: #34495e; /* Grey color */
   font-family: Georgia, serif;
   font-size: 16px;
   line-height: 1.6;
   margin-bottom: 15px;
```

```
}

/* Styling for div tag */
div {
  background-color: #ecf0f1; /* Light grey background */
  border: 1px solid #bdc3c7; /* Border color */
  padding: 15px;
  margin-bottom: 20px;
  border-radius: 5px;
}

/* Styling for span tag */
span {
  color: #e74c3c; /* Red color */
  font-weight: bold;
}

/* Styling for time tag */
time {
  color: #9b59b6; /* Purple color */
  font-style: italic;
}

/* Styling for img tag */
img {
  max-width: 100%;
  height: auto;
  border: 2px solid #3498db; /* Blue border */
  border-radius: 10px;
}

/* Styling for a tag */
a {
  color: #2980b9; /* Blue color */
  text-decoration: none;
  font-weight: bold;
}

a:hover {
  text-decoration: underline;
  color: #1f618d; /* Darker blue on hover */
}
```

Explanation of the CSS Styles and Selectors:

1. **Universal Selector (`*`)**:
   - Applies basic margin, padding, and box-sizing to all elements, ensuring consistency across different elements.
2. **Element Selector (`h2`, `h3`, `hr`, `p`, `div`, `span`, `time`, `img`, `a`)**:
   - Targets specific HTML elements to apply general styling.
3. **`h2`**:
   - Sets color, font-family, font-size, and margin for `h2` headers.
4. **`h3`**:
   - Applies color, font-family, font-size, and text transformation to `h3` headers.
5. **`hr`**:
   - Styles horizontal rules with background color, height, and margin.
6. **`p`**:
   - Defines color, font-family, font-size, line-height, and margin for paragraphs.
7. **`div`**:
   - Adds background color, border, padding, margin, and border-radius to `div` elements.
8. **`span`**:
   - Styles `span` elements with color and font-weight.
9. **`time`**:
   - Applies color and font-style to `time` elements.
10. **`img`**:
    - Sets maximum width, auto height, border, and border-radius for images, ensuring responsiveness.
11. **`a`**:
    - Styles links with color, removes underline by default, and adds bold font weight.
    - `a:hover` pseudo-class changes the link appearance when hovered over, adding underline and changing the color.

Applying the Styles

To use these styles in an HTML document, link the `style.css` file in the `<head>` section of your HTML file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Styled Page</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h2>Heading Level 2</h2>
  <h3>Heading Level 3</h3>
  <hr>
  <p>This is a paragraph. It has a specific font, color, and line-height.</p>
```

```
    <div>
        <p>This is a paragraph inside a div. The div has a background color and border.</p>
        <span>This is a span element with different styling.</span>
    </div>
    <time datetime="2024-08-27">August 27, 2024</time>
    <img src="example.jpg" alt="Example Image">
    <a href="https://www.example.com">Visit Example.com</a>
</body>
</html>
```

Output

Program 4

4. Develop HTML page named as "registration.html" having variety of HTML input elements with background colors, table for alignment & provide font colors & size using CSS styles.

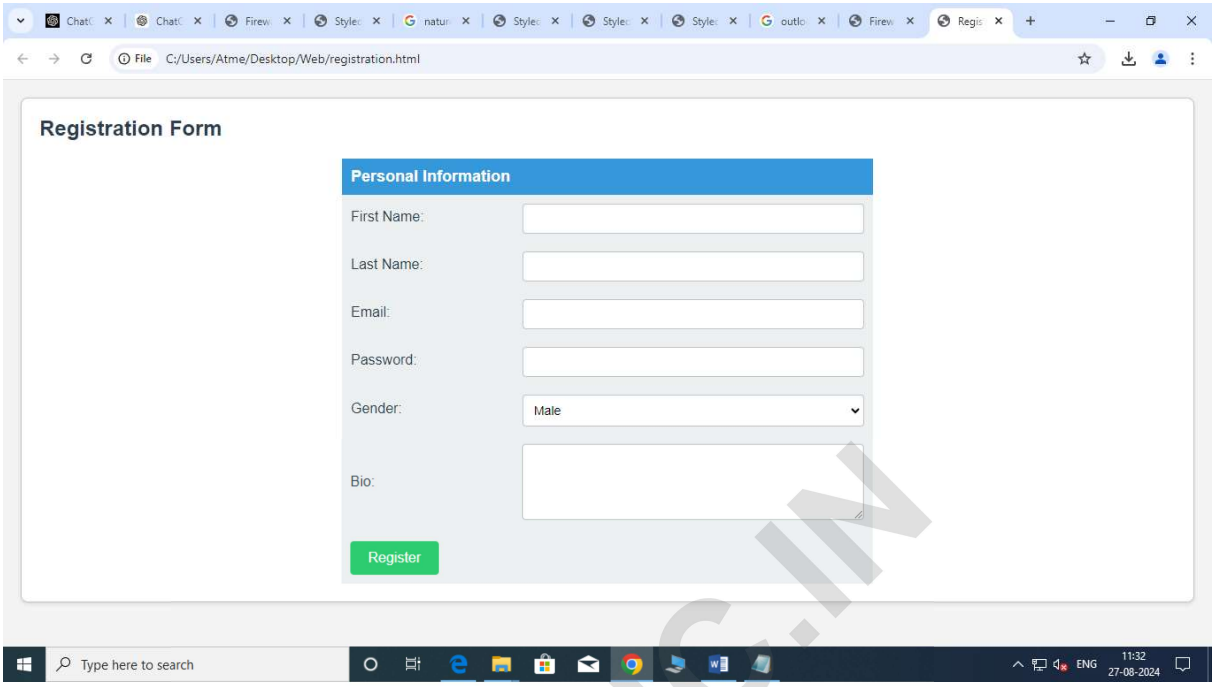
Program

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Form</title>
  <link rel="stylesheet" href="style.css">
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 20px;
    }
    table {
      width: 100%;
      max-width: 600px;
      margin: 0 auto;
      border-collapse: collapse;
    }
    th, td {
      padding: 10px;
      text-align: left;
    }
    th {
      background-color: #3498db;
      color: #fff;
      font-size: 18px;
    }
    td {
      background-color: #ecf0f1;
    }
    label {
      display: block;
      margin-bottom: 5px;
      font-size: 16px;
      color: #2c3e50;
    }
    input[type="text"],
    input[type="email"],
```

```
    input[type="password"],
    select,
    textarea {
      width: 100%;
      padding: 8px;
      border: 1px solid #bdc3c7;
      border-radius: 4px;
      font-size: 14px;
    }
    input[type="submit"] {
      background-color: #2ecc71;
      color: #fff;
      border: none;
      padding: 10px 20px;
      border-radius: 4px;
      font-size: 16px;
      cursor: pointer;
    }
    input[type="submit"]:hover {
      background-color: #27ae60;
    }
    .form-container {
      background-color: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    }
    .form-title {
      font-size: 24px;
      color: #2c3e50;
      margin-bottom: 20px;
    }
  </style>
</head>
<body>
  <div class="form-container">
    <h1 class="form-title">Registration Form</h1>
    <form action="#" method="post">
      <table>
        <tr>
          <th colspan="2">Personal Information</th>
        </tr>
        <tr>
          <td><label for="first-name">First Name:</label></td>
          <td><input type="text" id="first-name" name="first-name" required></td>
        </tr>
        <tr>
          <td><label for="last-name">Last Name:</label></td>
```

```
            <td><input type="text" id="last-name" name="last-name" required></td>
          </tr>
          <tr>
            <td><label for="email">Email:</label></td>
            <td><input type="email" id="email" name="email" required></td>
          </tr>
          <tr>
            <td><label for="password">Password:</label></td>
            <td><input type="password" id="password" name="password" required></td>
          </tr>
          <tr>
            <td><label for="gender">Gender:</label></td>
            <td>
              <select id="gender" name="gender">
                <option value="male">Male</option>
                <option value="female">Female</option>
                <option value="other">Other</option>
              </select>
            </td>
          </tr>
          <tr>
            <td><label for="bio">Bio:</label></td>
            <td><textarea id="bio" name="bio" rows="4"></textarea></td>
          </tr>
          <tr>
            <td colspan="2">
              <input type="submit" value="Register">
            </td>
          </tr>
        </table>
      </form>
    </div>
</body>
</html>
```

Output



Explanation of the HTML and CSS:

1. **HTML Structure**:
   - The form is placed inside a `div` with the class `form-container` for styling.
   - A table is used for layout alignment, with each row containing labels and corresponding input fields.
   - Various input types (`text`, `email`, `password`, `select`, `textarea`) are included to demonstrate different form elements.
2. **CSS Styles**:
   - **`body`**: Sets a background color and padding for the entire page.
   - **`table`**: Centers the table and sets its width. Uses `border-collapse` to ensure borders are collapsed into a single line.
   - **`th`**: Styles table headers with a background color and white text.
   - **`td`**: Styles table data cells with a light grey background.
   - **`label`**: Styles labels with font size and color.
   - **`input, select, textarea`**: Applies styles to form elements including padding, border, and font size.
   - **`input[type="submit"]`**: Styles the submit button with a green background, white text, and hover effect.
   - **`.form-container`**: Adds padding, background color, border radius, and box shadow to the form container for a card-like appearance.
   - **`.form-title`**: Styles the form title with font size and color.

Program 5

5. Develop HTML page named as "newspaper.html" having variety of HTML semantic elements with background colors, text-colors & size for figure, table, aside, section, article, header, footer… etc.

Program

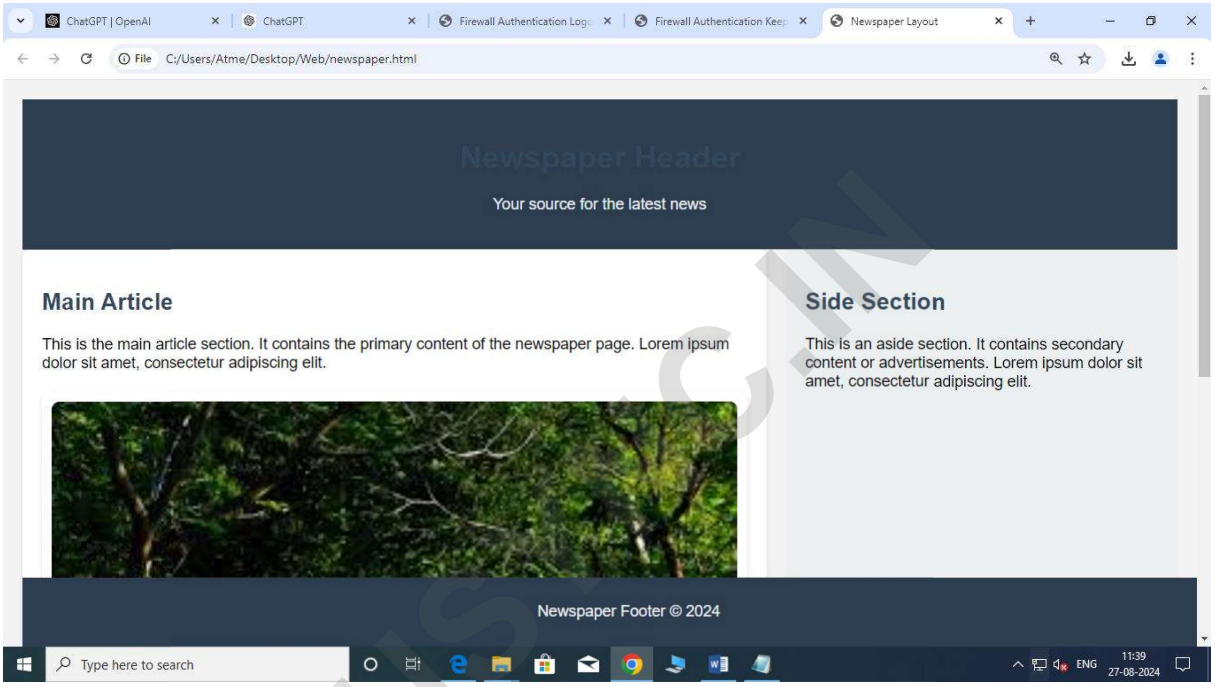
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Newspaper Layout</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 20px;
    }
    header {
      background-color: #2c3e50;
      color: #ecf0f1;
      padding: 20px;
      text-align: center;
    }
    footer {
      background-color: #2c3e50;
      color: #ecf0f1;
      padding: 10px;
      text-align: center;
      position: fixed;
      bottom: 0;
      width: 100%;
    }
    main {
      display: flex;
      gap: 20px;
      margin-bottom: 60px; /* To ensure footer does not overlap content */
    }
    article {
      flex: 2;
      background-color: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    }
```

```
    aside {
      flex: 1;
      background-color: #ecf0f1;
      padding: 20px;
      border-radius: 8px;
    }
    section {
      margin-bottom: 20px;
    }
    h1, h2, h3 {
      color: #34495e;
    }
    table {
      width: 100%;
      border-collapse: collapse;
      margin-top: 20px;
    }
    table, th, td {
      border: 1px solid #bdc3c7;
    }
    th {
      background-color: #3498db;
      color: #fff;
      padding: 10px;
    }
    td {
      background-color: #fff;
      padding: 10px;
      text-align: left;
    }
    figure {
      margin: 0;
      padding: 0;
    }
    figcaption {
      font-size: 14px;
      color: #7f8c8d;
      text-align: center;
    }
    .figure-container {
      background-color: #fff;
      padding: 10px;
      border-radius: 8px;
      box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
      margin: 20px 0;
    }
  </style>
</head>
```

```
<body>
  <header>
    <h1>Newspaper Header</h1>
    <p>Your source for the latest news</p>
  </header>

  <main>
    <article>
      <section>
        <h2>Main Article</h2>
        <p>This is the main article section. It contains the primary content of the newspaper page. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
        <figure class="figure-container">
          <img src="example.jpg" alt="Example Image" style="width: 100%; border-radius: 8px;">
          <figcaption>Figure 1: Example image caption.</figcaption>
        </figure>
        <table>
          <caption>Sample Table</caption>
          <thead>
            <tr>
              <th>Header 1</th>
              <th>Header 2</th>
              <th>Header 3</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td>Data 1</td>
              <td>Data 2</td>
              <td>Data 3</td>
            </tr>
            <tr>
              <td>Data 4</td>
              <td>Data 5</td>
              <td>Data 6</td>
            </tr>
          </tbody>
        </table>
      </section>
    </article>

    <aside>
      <h2>Side Section</h2>
      <p>This is an aside section. It contains secondary content or advertisements. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    </aside>
  </main>
```

```
  <footer>
    <p>Newspaper Footer © 2024</p>
  </footer>
</body>
</html>
```

Output



Explanation of the HTML and CSS:

1. **HTML Structure**:
   - **`<header>`**: Contains the main heading and subtitle of the newspaper.
   - **`<footer>`**: Provides footer information, fixed at the bottom of the page.
   - **`<main>`**: Uses flexbox to layout the `article` and `aside` side by side.
   - **`<article>`**: Represents the main content area of the page.
   - **`<aside>`**: Contains secondary content or sidebar information.
   - **`<section>`**: Used within the `article` to group related content.
   - **`<figure>`** and **`<figcaption>`**: Used to include an image with a caption.
   - **`<table>`**: Contains tabular data with headers and a caption.

2. **CSS Styles**:
   - **`body`**: Sets a background color, font, and padding for the entire page.
   - **`header`** and **`footer`**: Define background colors, text colors, and padding. The footer is positioned fixed at the bottom of the page.
   - **`main`**: Utilizes flexbox to create a responsive layout for the main content and sidebar.

- `article` and `aside`: Style the primary content and sidebar with background colors, padding, and shadows.
- `table`: Styles the table with borders, background colors for headers, and padding.
- `figure` and `figcaption`: Apply styling to the figure and caption, including background color and text color.

Program 6

6. Apply HTML, CSS and JavaScript to design a simple calculator to perform the following operations: sum, product, difference, remainder, quotient, power, square-root and square.

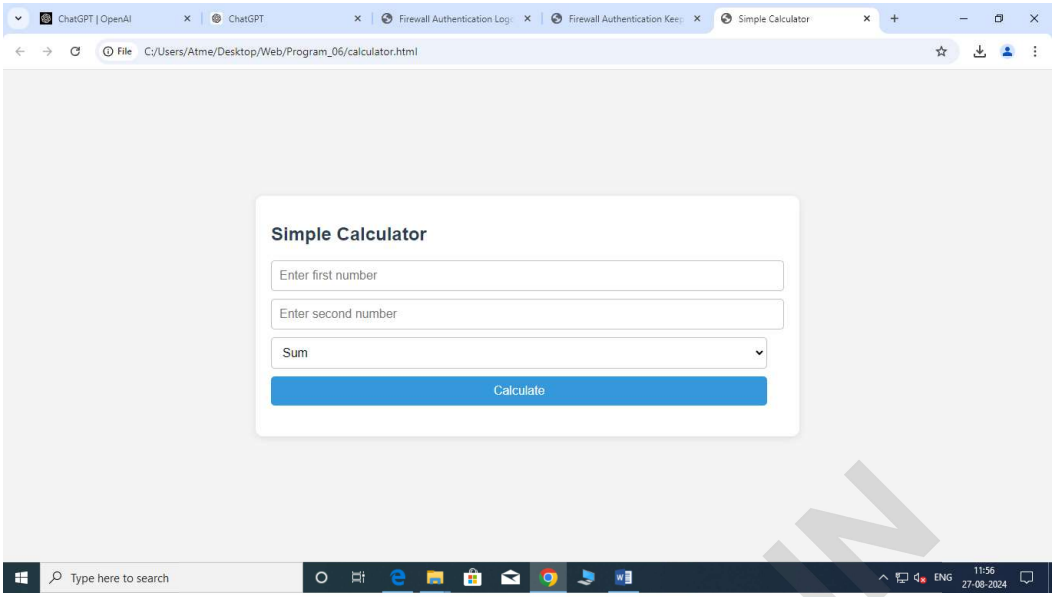
Program

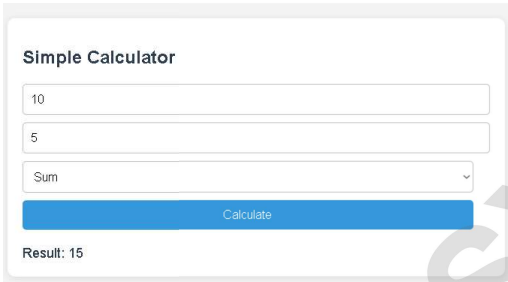
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Simple Calculator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background-color: #f4f4f4;
      margin: 0;
    }
    .calculator {
      background-color: #fff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    }
    .calculator h1 {
      font-size: 24px;
      color: #2c3e50;
      margin-bottom: 20px;
    }
    .calculator input,
    .calculator select,
    .calculator button {
      margin-bottom: 10px;
      padding: 10px;
      border: 1px solid #bdc3c7;
      border-radius: 5px;
      font-size: 16px;
      width: calc(100% - 22px); /* Adjust for padding and border */
    }
    .calculator button {
      background-color: #3498db;
      color: #fff;
      border: none;
```

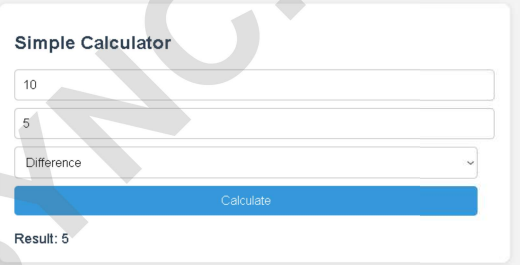
```
        cursor: pointer;
      }
      .calculator button:hover {
        background-color: #2980b9;
      }
      .calculator .result {
        font-size: 18px;
        color: #2c3e50;
        margin-top: 10px;
      }
    </style>
</head>
<body>
    <div class="calculator">
      <h1>Simple Calculator</h1>
      <input type="number" id="num1" placeholder="Enter first number">
      <input type="number" id="num2" placeholder="Enter second number">
      <select id="operation">
        <option value="sum">Sum</option>
        <option value="difference">Difference</option>
        <option value="product">Product</option>
        <option value="quotient">Quotient</option>
        <option value="remainder">Remainder</option>
        <option value="power">Power</option>
        <option value="sqrt">Square Root</option>
        <option value="square">Square</option>
      </select>
      <button onclick="calculate()">Calculate</button>
      <div class="result" id="result"></div>
    </div>

    <script>
      function calculate() {
        // Get values from inputs
        const num1 = parseFloat(document.getElementById('num1').value);
        const num2 = parseFloat(document.getElementById('num2').value);
        const operation = document.getElementById('operation').value;

        // Result variable
        let result;

        // Perform calculation based on selected operation
        switch (operation) {
          case 'sum':
            result = num1 + num2;
            break;
          case 'difference':
            result = num1 - num2;
```

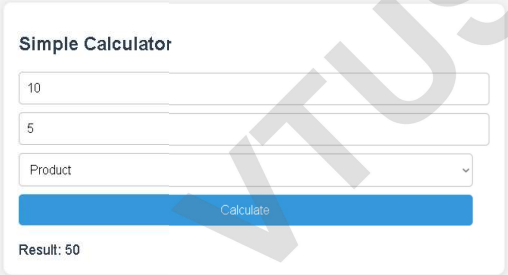
```
            break;
        case 'product':
            result = num1 * num2;
            break;
        case 'quotient':
            result = num1 / num2;
            break;
        case 'remainder':
            result = num1 % num2;
            break;
        case 'power':
            result = Math.pow(num1, num2);
            break;
        case 'sqrt':
            result = Math.sqrt(num1);
            break;
        case 'square':
            result = Math.pow(num1, 2);
            break;
        default:
            result = 'Invalid operation';
    }

    // Display result
    document.getElementById('result').textContent = 'Result: ' + result;
  }
 </script>
</body>
</html>
```

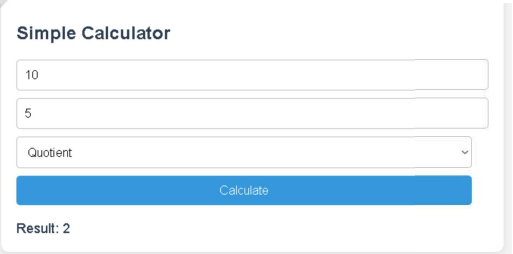
Output

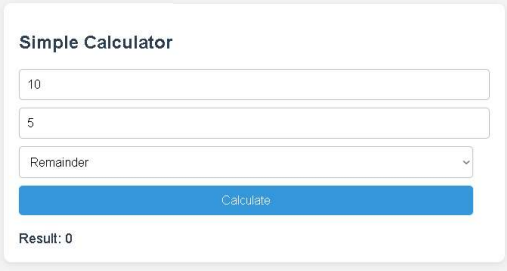


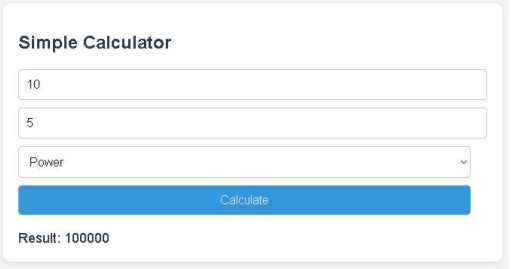


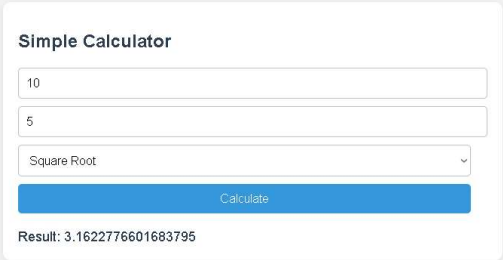


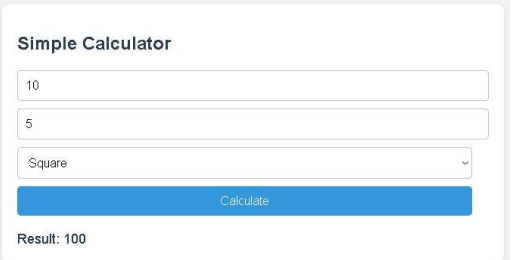












Explanation:

1. **HTML**:
   - The HTML structure includes:
     - Two input fields for numbers.
     - A `<select>` dropdown to choose the operation.
     - A `<button>` to trigger the calculation.
     - A `<div>` to display the result.
2. **CSS**:
   - Styles are applied to center the calculator on the page, style the form elements, and add hover effects to the button.
   - The `.calculator` class styles the container, adding padding, border-radius, and box-shadow for a clean look.
3. **JavaScript**:
   - The `calculate` function retrieves the values from the input fields and the selected operation from the dropdown.
   - It performs the calculation based on the chosen operation using a `switch` statement.
   - The result is then displayed in the `<div>` with the `id="result"`.

This simple calculator performs all the specified operations, and the JavaScript handles the logic for each operation based on user input.

Program 7

7. Develop JavaScript program (with HTML/CSS) for:
   a) Converting JSON text to JavaScript Object
   b) Convert JSON results into a date
   c) Converting from JSON to CSV and CSV to JSON
   d) Create hash from string using crypto.createHash() method

Program

```
<!DOCTYPE html>
<html lang="en">
        <head>
          <meta charset="UTF-8">
          <meta name="viewport" content="width=device-width, initial-scale=1.0">
          <title>JSON and CSV Converter</title>
          <style>
            body {
              font-family: Arial, sans-serif;
              margin: 20px;
            }
            .container {
              margin-bottom: 20px;
            }
            textarea, input[type="text"], button {
              display: block;
              width: 100%;
              margin-bottom: 10px;
              padding: 8px;
              box-sizing: border-box;
            }
            button {
              background-color: #4CAF50;
              color: white;
              border: none;
              cursor: pointer;
            }
            button:hover {
              background-color: #45a049;
            }
            .result {
              white-space: pre-wrap;
              background: #f4f4f4;
              padding: 10px;
            }
          </style>
        </head>
        <body>
```

```
    <div class="container">
      <h2>Convert JSON to JavaScript Object</h2>
      <textarea id="jsonText" placeholder='Enter JSON here'></textarea>
      <button onclick="convertJsonToObject()">Convert JSON</button>
      <div id="jsonObject" class="result"></div>
    </div>

    <div class="container">
      <h2>Convert JSON to Date</h2>
      <input type="text" id="jsonDate" placeholder='Enter JSON Date string'>
      <button onclick="convertJsonToDate()">Convert to Date</button>
      <div id="dateResult" class="result"></div>
    </div>

    <div class="container">
      <h2>Convert JSON to CSV and CSV to JSON</h2>
      <textarea id="jsonCsvText" placeholder='Enter JSON for CSV conversion'></textarea>
      <button onclick="convertJsonToCsv()">JSON to CSV</button>
      <div id="csvResult" class="result"></div>

      <textarea id="csvJsonText" placeholder='Enter CSV text'></textarea>
      <button onclick="convertCsvToJson()">CSV to JSON</button>
      <div id="jsonResult" class="result"></div>
    </div>

    <div class="container">
      <h2>Create Hash from String</h2>
      <input type="text" id="stringInput" placeholder='Enter string to hash'>
      <button onclick="createHash()">Create Hash</button>
      <div id="hashResult" class="result"></div>
    </div>

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


JavaScript (app.js)

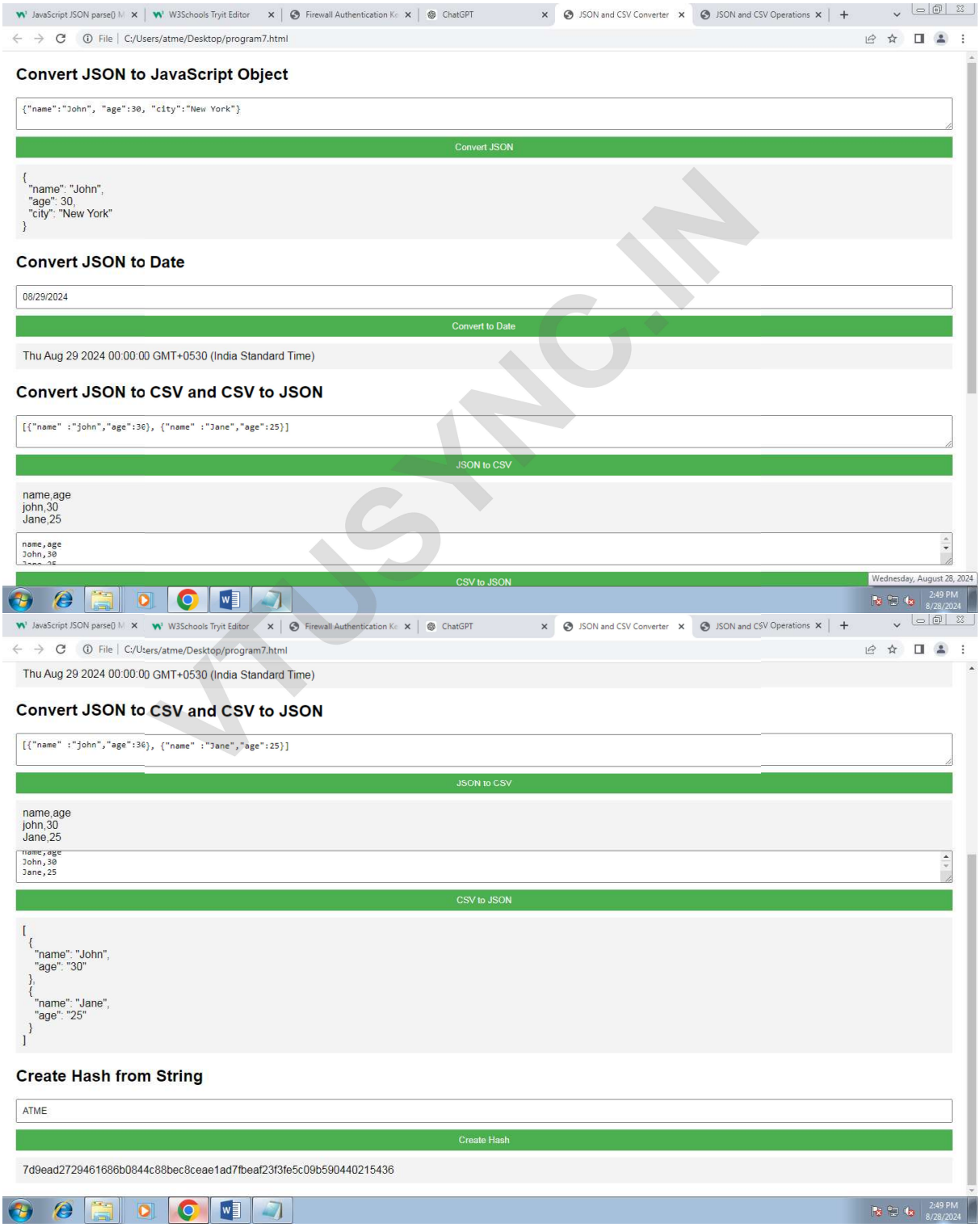
Now let's add the JavaScript code to handle each of the tasks.

```
}// Convert JSON text to JavaScript Object
function convertJsonToObject() {
  const jsonText = document.getElementById('jsonText').value;
  try {
    const obj = JSON.parse(jsonText);
    document.getElementById('jsonObject').textContent = JSON.stringify(obj, null, 2);
  } catch (e) {
```

```
        document.getElementById('jsonObject').textContent = 'Invalid JSON';
    }
}

// Convert JSON date string to Date
function convertJsonToDate() {
    const jsonDate = document.getElementById('jsonDate').value;
    try {
        const date = new Date(jsonDate);
        document.getElementById('dateResult').textContent = date.toString();
    } catch (e) {
        document.getElementById('dateResult').textContent = 'Invalid Date';
    }
}

// Convert JSON to CSV
function convertJsonToCsv() {
    const jsonText = document.getElementById('jsonCsvText').value;
    try {
        const data = JSON.parse(jsonText);
        const csv = Object.keys(data[0]).join(',') + '\n' +
                data.map(row => Object.values(row).join(',')).join('\n');
        document.getElementById('csvResult').textContent = csv;
    } catch (e) {
        document.getElementById('csvResult').textContent = 'Invalid JSON';
    }
}

// Convert CSV to JSON
function convertCsvToJson() {
    const csvText = document.getElementById('csvJsonText').value;
    const rows = csvText.split('\n').map(row => row.split(','));
    const headers = rows.shift();
    const json = rows.map(row => {
        let obj = {};
        row.forEach((value, index) => obj[headers[index]] = value);
        return obj;
    });
    document.getElementById('jsonResult').textContent = JSON.stringify(json, null, 2);
}

// Create hash from string
async function createHash() {
    const crypto = window.crypto || window.msCrypto; // For IE11
    const str = document.getElementById('stringInput').value;
    const encoder = new TextEncoder();
    const data = encoder.encode(str);
    const hashBuffer = await crypto.subtle.digest('SHA-256', data);
```

```
const hashArray = Array.from(new Uint8Array(hashBuffer));
const hashHex = hashArray.map(b => b.toString(16).padStart(2, '0')).join('');
document.getElementById('hashResult').textContent = hashHex;
```

Output



Explanation:

1. **Convert JSON to JavaScript Object**: The function `convertJsonToObject` parses the JSON text and displays it in a formatted way. If the JSON is invalid, an error message is shown.
2. **Convert JSON Date to JavaScript Date**: The function `convertJsonToDate` creates a JavaScript `Date` object from the JSON date string and displays it.
3. **Convert JSON to CSV**: The function `convertJsonToCsv` parses JSON data into a CSV format. It assumes that the input JSON is an array of objects.
4. **Convert CSV to JSON**: The function `convertCsvToJson` parses CSV data into JSON. It assumes the first row of the CSV contains headers.
5. **Create Hash from String**: The function `createHash` uses the Web Crypto API to generate a SHA-256 hash of the input string.

Program 8

8. a. Develop a PHP program (with HTML/CSS) to keep track of the number of visitors visiting the web page and to display this count of visitors, with relevant headings.
   b. Develop a PHP program (with HTML/CSS) to sort the student records which are stored in the database using selection sort.

Program 8a

Develop a PHP program (with HTML/CSS) to keep track of the number of visitors visiting the web page and to display this count of visitors, with relevant headings.

Step 1: Create the PHP Script

First, create a PHP file named `program_8a.php`. This script will handle the counting of visitors and display the count on the web page.

```
<?php
// Define the path to the file where the count will be stored
$countFile = 'count.txt';

// Check if the count file exists
if (!file_exists($countFile)) {
   // If the file doesn't exist, create it and initialize the count to 0
   file_put_contents($countFile, '0');
}

// Read the current count from the file
$count = (int)file_get_contents($countFile);

// Increment the count
$count++;

// Save the updated count back to the file
file_put_contents($countFile, $count);

// HTML output
?>
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Visitor Counter</title>
   <style>
      body {
         font-family: Arial, sans-serif;
         display: flex;
         justify-content: center;
         align-items: center;
```

```
        height: 100vh;
        background-color: #f4f4f4;
        margin: 0;
      }
      .container {
        text-align: center;
        background-color: #fff;
        padding: 20px;
        border-radius: 8px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
      }
      h1 {
        color: #333;
      }
      .count {
        font-size: 2em;
        color: #007bff;
      }
    </style>
</head>
<body>
    <div class="container">
      <h1>Visitor Counter</h1>
      <p class="count">You are visitor number: <?php echo $count; ?></p>
    </div>
</body>
</html>
```

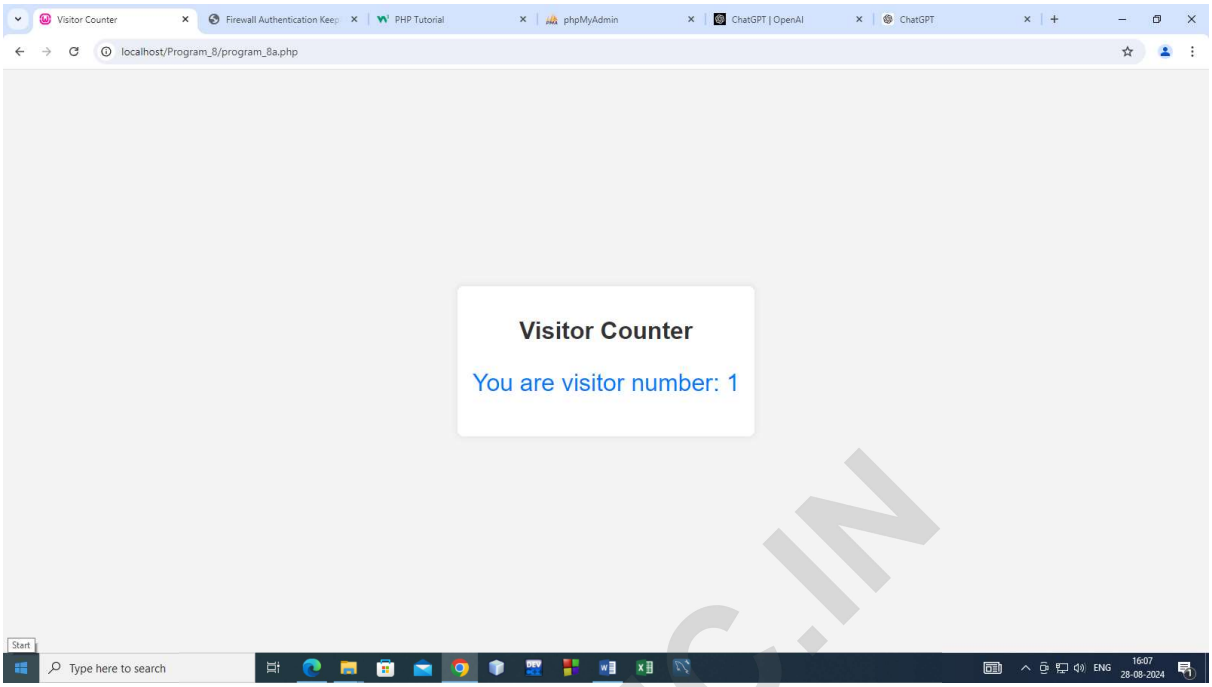

Step 2: Explanation

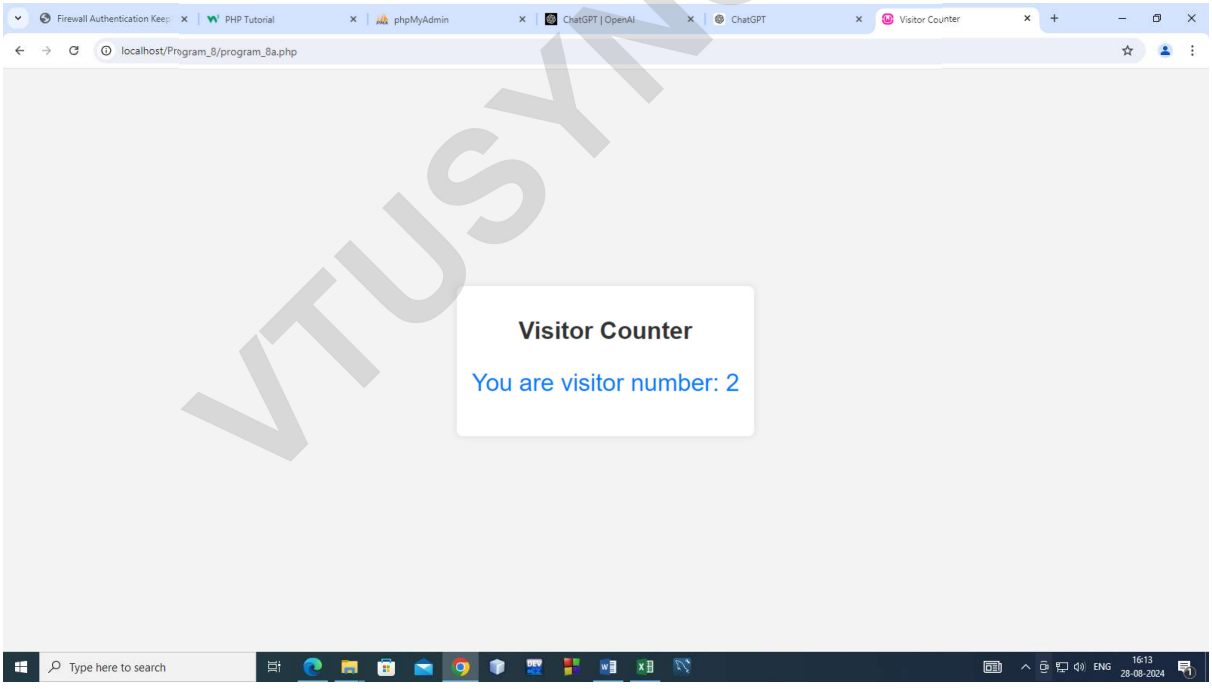
1. **File Handling**: The PHP script uses `count.txt` to keep track of the number of visitors. If this file does not exist, it creates it and initializes the count to 0. Then, it reads the current count, increments it, and writes the updated count back to the file.
2. **HTML/CSS**:
   - **HTML**: The page has a simple structure with a heading and a paragraph that displays the visitor count.
   - **CSS**: Basic styles are applied to center the content and make the page look clean. The container has a box shadow for a subtle effect, and the visitor count is styled to stand out.

Step 3: Upload and Test

1. **Upload Files**: Place `program_8a.php` on your web server. Ensure that `count.txt` is writable by the web server (check permissions if needed).
2. **Access Your Page**: Open your web browser and navigate to the location where `program_8a.php` is hosted. You should see the visitor counter and the count will increment with each page load.

Output





Program 8b

Develop a PHP program (with HTML/CSS) to sort the student records which are stored in the database using selection sort.

Steps

1. **Set up your database**: You'll need a MySQL database with a table for student records. For simplicity, let's assume you have a database named `web_lab_students` and a table named `students` with fields `Stu_name, USN` and `Sem`.
2. **Create the PHP script**: This script will connect to the database, fetch student records, sort them using selection sort, and then display the results.
3. **Write the HTML/CSS**: This part will handle the presentation of the sorted data.

1. Database Setup

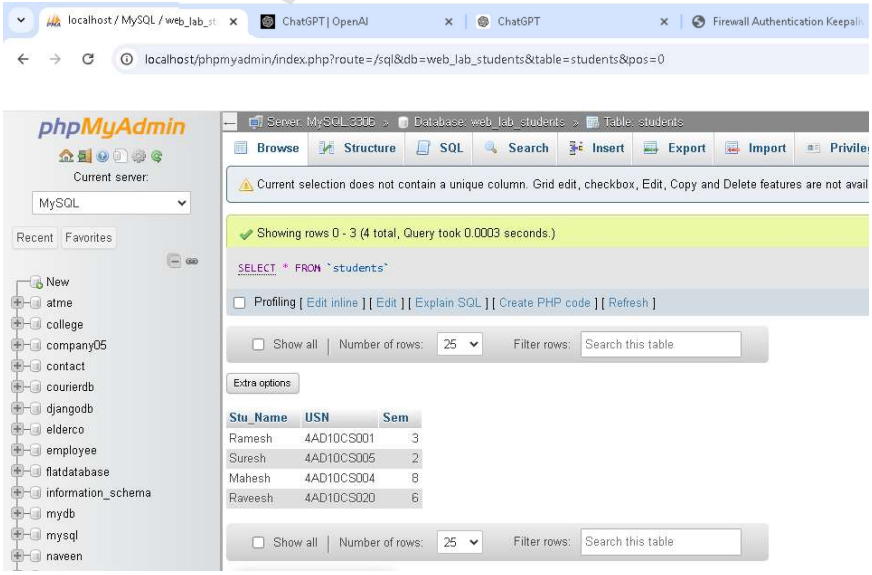
Assuming your database `web_lab_students` and table `students` are already created, here is an example of how the table might look:

CREATE DATABASE `web_lab_students`;

USE `web_lab_students`;

CREATE TABLE `students` (
`Stu_Name` VARCHAR(100),
`USN` VARCHAR(20),
`Sem` INT);

INSERT INTO students (Stu_Name, USN, Sem) VALUES('Ramesh', '4AD10CS001',3)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Suresh', '4AD10CS005',2)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Mahesh', '4AD10CS004',8)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Raveesh', '4AD10CS020',6)



2. PHP Script (with Selection Sort)

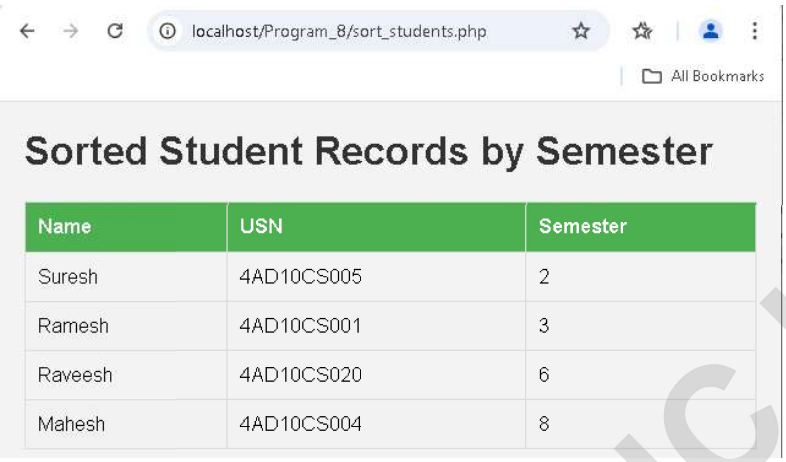
Create a file named `sort_students.php` with the following content:

```
<?php
// Database configuration
$host = 'localhost';
$dbname = 'web_lab_students';
$username = 'root'; // Replace with your MySQL username
$password = 'root'; // Replace with your MySQL password

// Create a new PDO instance
try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    die("Could not connect to the database: " . $e->getMessage());
}

// Fetch student records from the database
$sql = 'SELECT * FROM students';
$stmt = $pdo->query($sql);
$students = $stmt->fetchAll(PDO::FETCH_ASSOC);

// Function to perform selection sort
function selectionSort(&$array) {
    $n = count($array);
    for ($i = 0; $i < $n - 1; $i++) {
        $minIndex = $i;
        for ($j = $i + 1; $j < $n; $j++) {
            if ($array[$j]['Sem'] < $array[$minIndex]['Sem']) {
                $minIndex = $j;
            }
        }
        if ($minIndex != $i) {
            $temp = $array[$i];
            $array[$i] = $array[$minIndex];
            $array[$minIndex] = $temp;
        }
    }
}

// Sort students by semester using selection sort
selectionSort($students);
?>
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sorted Student Records</title>
    <style>
      body {
        font-family: Arial, sans-serif;
        margin: 20px;
        background-color: #f4f4f4;
      }
      table {
        width: 100%;
        border-collapse: collapse;
      }
      th, td {
        padding: 10px;
        border: 1px solid #ddd;
        text-align: left;
      }
      th {
        background-color: #4CAF50;
        color: white;
      }
      tr:nth-child(even) {
        background-color: #f2f2f2;
      }
      h1 {
        color: #333;
      }
    </style>
  </head>
  <body>
    <h1>Sorted Student Records by Semester</h1>
    <table>
      <thead>
        <tr>
          <th>Name</th>
          <th>USN</th>
          <th>Semester</th>
        </tr>
      </thead>
      <tbody>
        <?php foreach ($students as $student): ?>
          <tr>
            <td><?php echo htmlspecialchars($student['Stu_Name']); ?></td>
            <td><?php echo htmlspecialchars($student['USN']); ?></td>
            <td><?php echo htmlspecialchars($student['Sem']); ?></td>
          </tr>
        <?php endforeach; ?>
```

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

Output



Explanation

1. **Database Connection**: Establish a connection to the MySQL database using PDO.
2. **Fetching Data**: Retrieve all student records from the `students` table.
3. **Selection Sort Function**: Implement the selection sort algorithm to sort students by their semester (`sem`).
4. **HTML and CSS**: Display the sorted student records in an HTML table, styled with basic CSS.

3. Running the Script

- Make sure your MySQL server is running, and PHP is properly configured.
- Place the `sort_students.php` file in your web server's root directory.
- Access the script via your browser, for example, `http://localhost/sort_students.php`.

This script will display student records sorted by the `sem` field using the selection sort algorithm. Adjust database credentials and table schema as needed.

Program 9

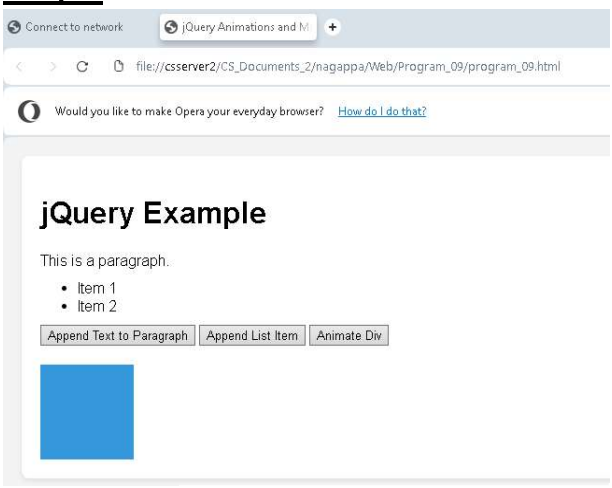
9. Develop jQuery script (with HTML/CSS) for:
   a. Appends the content at the end of the existing paragraph and list.
   b. Change the state of the element with CSS style using animate() method
   c. Change the color of any div that is animated.

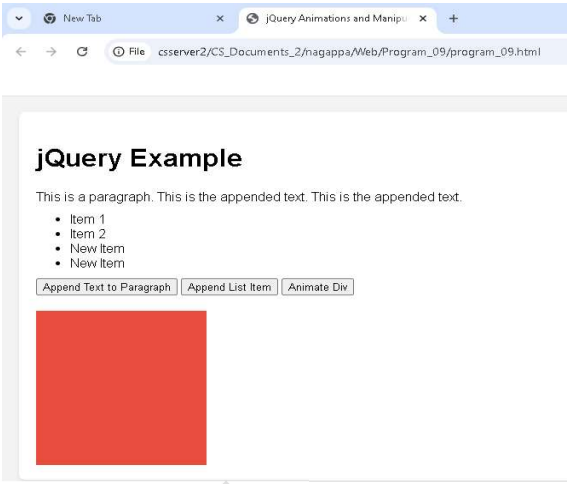
Program

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>jQuery Animations and Manipulations</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 20px;
      background-color: #f4f4f4;
    }
    .container {
      background-color: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    }
    p, ul {
      margin: 10px 0;
    }
    .animated-div {
      width: 100px;
      height: 100px;
      background-color: #3498db;
      margin-top: 20px;
      transition: background-color 0.5s;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>jQuery Example</h1>

    <!-- Paragraph and List to append content to -->
    <p id="paragraph">This is a paragraph.</p>
    <ul id="list">
```

```
        <li>Item 1</li>
        <li>Item 2</li>
      </ul>

      <!-- Buttons for jQuery actions -->
      <button id="appendText">Append Text to Paragraph</button>
      <button id="appendListItem">Append List Item</button>
      <button id="animateDiv">Animate Div</button>

      <!-- Div to animate -->
      <div id="animatedDiv" class="animated-div"></div>
    </div>

    <!-- jQuery Library -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
      $(document).ready(function() {
        // Append content to paragraph and list
        $('#appendText').click(function() {
          $('#paragraph').append(' This is the appended text.');
        });

        $('#appendListItem').click(function() {
          $('#list').append('<li>New Item</li>');
        });

        // Animate div
        $('#animateDiv').click(function() {
          $('#animatedDiv').animate({
            width: '200px',
            height: '200px'
          }, 1000, function() {
            // Change color after animation completes
            $(this).css('background-color', '#e74c3c');
          });
        });
      });
    </script>
  </body>
</html>
```

Output





Explanation:

1. **HTML**:
   - Contains a paragraph and a list to which content can be appended.
   - Includes buttons for appending text, appending list items, and animating a `div`.
   - The `div` with the `id="animatedDiv"` is styled and animated.

2. **CSS**:
   - Basic styling for the body, container, paragraph, and list.
   - The `.animated-div` class provides initial styling and a transition effect for background color changes.

3. **jQuery**:
   - **Append Content**:
     - `$('#appendText').click()`: Appends text to the existing paragraph when the button is clicked.
     - `$('#appendListItem').click()`: Appends a new list item to the existing list when the button is clicked.
   - **Animate Div**:
     - `$('#animateDiv').click()`: Animates the size of the `div` when the button is clicked. After the animation completes, the background color of the `div` changes to red.

This script demonstrates basic jQuery functionalities for manipulating and animating HTML elements. You can save the above code into an `index.html` file and open it in a web browser to see it in action.

Program 10

10. Develop a JavaScript program with Ajax (with HTML/CSS) for:
    a. Use ajax() method (without Jquery) to add the text content from the text file by sending ajax request.
    b. Use ajax() method (with Jquery) to add the text content from the text file by sending ajax request.
    c. Illustrate the use of getJSON() method in jQuery
    d. Illustrate the use of parseJSON() method to display JSON values.

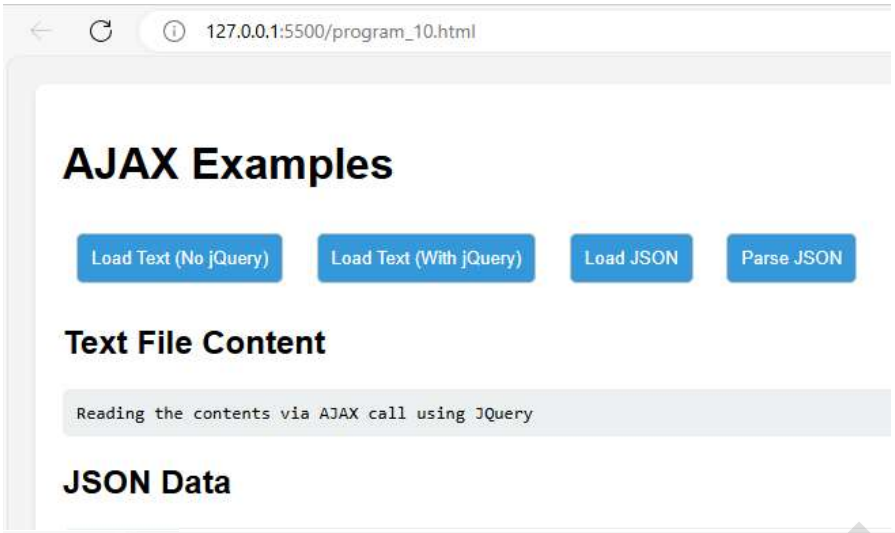
Program

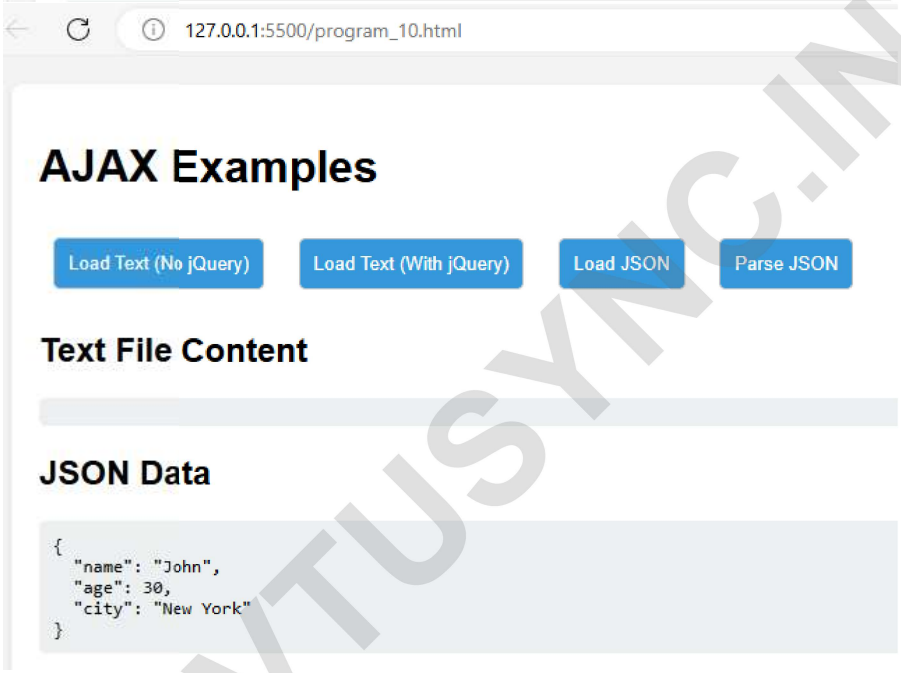
```
<!DOCTYPE html>
<html lang="en">
        <head>
          <meta charset="UTF-8">
          <meta name="viewport" content="width=device-width, initial-scale=1.0">
          <title>AJAX with and without jQuery</title>
          <style>
            body {
              font-family: Arial, sans-serif;
              margin: 20px;
              background-color: #f4f4f4;
            }
            .container {
              background-color: #fff;
              padding: 20px;
              border-radius: 8px;
              box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
            }
            button {
              margin: 10px;
              padding: 10px;
              border: 1px solid #bdc3c7;
              border-radius: 5px;
              background-color: #3498db;
              color: #fff;
              cursor: pointer;
            }
            button:hover {
              background-color: #2980b9;
            }
            pre {
              background-color: #ecf0f1;
              padding: 10px;
              border-radius: 5px;
              white-space: pre-wrap;
```

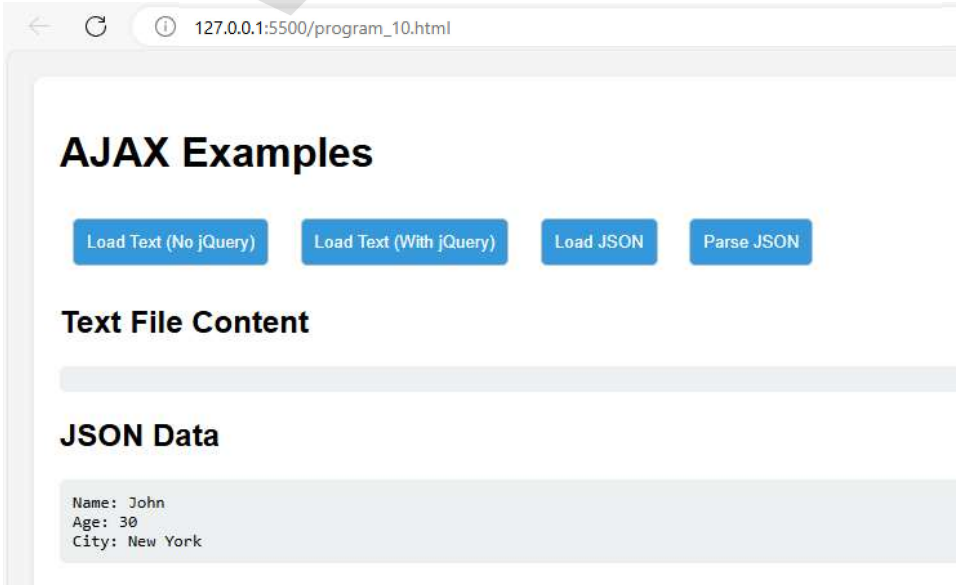
```
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>AJAX Examples</h1>

        <!-- Buttons for AJAX requests -->
        <button id="loadTextWithoutJQuery">Load Text (No jQuery)</button>
        <button id="loadTextWithJQuery">Load Text (With jQuery)</button>
        <button id="loadJson">Load JSON</button>
        <button id="parseJson">Parse JSON</button>

        <!-- Output areas -->
        <h2>Text File Content</h2>
        <pre id="textOutput"></pre>

        <h2>JSON Data</h2>
        <pre id="jsonOutput"></pre>
    </div>

    <!-- jQuery Library -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

    <!-- JavaScript -->
    <script>
        // Function to load text content using vanilla JavaScript
        document.getElementById('loadTextWithoutJQuery').addEventListener('click',
function() {
            var xhr = new XMLHttpRequest();
            xhr.open('GET', 'sample.txt', true);
            xhr.onreadystatechange = function() {
                if (xhr.readyState === 4 && xhr.status === 200) {
                    document.getElementById('textOutput').textContent = xhr.responseText;
                }
            };
            xhr.send();
        });

        // Function to load text content using jQuery
        $('#loadTextWithJQuery').click(function() {
            $.ajax({
                url: 'sample.txt',
                method: 'GET',
                success: function(data) {
                    $('#textOutput').text(data);
                },
                error: function() {
```

```
                $('#textOutput').text('Error loading text file.');
            }
        });
    });

    // Function to load JSON data using jQuery's getJSON method
    $('#loadJson').click(function() {
        $.getJSON('data.json', function(data) {
            $('#jsonOutput').text(JSON.stringify(data, null, 2));
        }).fail(function() {
            $('#jsonOutput').text('Error loading JSON file.');
        });
    });

    // Function to parse JSON data using jQuery's parseJSON method
    $('#parseJson').click(function() {
        var jsonString = '{"name": "John", "age": 30, "city": "New York"}';
        var jsonObject = $.parseJSON(jsonString);
        $('#jsonOutput').text('Name: ' + jsonObject.name + '\nAge: ' + jsonObject.age +
'\nCity: ' + jsonObject.city);
    });
  </script>
 </body>
</html>
```

Output


127.0.0.1:5500/program_10.html
AJAX Examples
Load Text (No jQuery)
Load Text (With jQuery)
Load JSON
Parse JSON
Text File Content
Reading the contents via AJAX call using JQuery
JSON Data


127.0.0.1:5500/program_10.html
AJAX Examples
Load Text (No jQuery)
Load Text (With jQuery)
Load JSON
Parse JSON
Text File Content
JSON Data
{
"name": "John",
"age": 30,
"city": "New York"
}


127.0.0.1:5500/program_10.html
AJAX Examples
Load Text (No jQuery)
Load Text (With jQuery)
Load JSON
Parse JSON
Text File Content
JSON Data
Name: John
Age: 30
City: New York