Scenario 1: Responsive Navigation Bar

Task: Create a responsive navigation bar that collapses into a hamburger menu on smaller screens.

Solution:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Responsive Navigation Bar</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }
        .navbar {
            display: flex;
            justify-content: space-between;
            align-items: center;
            background-color: #333;
            padding: 10px;
        }
        .navbar a {
            color: white;
            text-decoration: none;
            padding: 10px;
        }
        .navbar .menu {
            display: none;
            flex-direction: column;
        }
        .navbar .menu a {
            text-align: center;
        }
        .navbar .hamburger {
            display: none;
            cursor: pointer;
        }
        @media (max-width: 768px) {
            .navbar .menu {
                display: none;
                flex-direction: column;
            }
            .navbar .menu.active {
                display: flex;
            }
            .navbar .hamburger {
                display: block;
                color: white;
            }
        }
    </style>
</head>
<body>
    <div class="navbar">
        <a href="#">Logo</a>
        <div class="hamburger" onclick="toggleMenu()">☰</div>
        <div class="menu">
            <a href="#">Home</a>
            <a href="#">About</a>
            <a href="#">Services</a>
            <a href="#">Contact</a>
        </div>
    </div>
    <script>
        function toggleMenu() {
            document.querySelector('.menu').classList.toggle('active');
        }
    </script>
</body>
</html>
```

Scenario 2: Flexbox Centering
Task: Center a div both horizontally and vertically within its parent using Flexbox.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Flexbox Centering</title>
    <style>
        .parent {
            display: flex;
            justify-content: center;
            align-items: center;
```

```
      height: 100vh;
      background-color: #f0f0f0;
    }
    .child {
      width: 200px;
      height: 200px;
      background-color: #333;
      color: white;
      display: flex;
      justify-content: center;
      align-items: center;
    }
  </style>
</head>
<body>
  <div class="parent">
    <div class="child">Centered</div>
  </div>
</body>
</html>
```

Scenario 3: CSS Grid Layout
Task: Create a 3-column grid layout that becomes 1-column on smaller screens.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
```

```
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Grid Layout</title>
  <style>
    .grid-container {
      display: grid;
      grid-template-columns: repeat(3, 1fr);
      gap: 10px;
    }
    .grid-item {
      background-color: #333;
      color: white;
      padding: 20px;
      text-align: center;
    }
    @media (max-width: 768px) {
      .grid-container {
        grid-template-columns: 1fr;
      }
    }
  </style>
</head>
<body>
  <div class="grid-container">
    <div class="grid-item">Item 1</div>
    <div class="grid-item">Item 2</div>
    <div class="grid-item">Item 3</div>
  </div>
</body>
</html>
```

Scenario 4: CSS Transitions

Task: Create a button that changes color with a smooth transition when hovered.

Solution:

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Transitions</title>
  <style>
    .button {
      background-color: #333;
      color: white;
      padding: 10px 20px;
      border: none;
      cursor: pointer;
      transition: background-color 0.3s ease;
    }
    .button:hover {
      background-color: #555;
    }
  </style>
</head>
<body>
  <button class="button">Hover me</button>
</body>
```

</html>
~~~

Scenario 5: CSS Animations
Task: Create a CSS animation that makes an element fade in and out.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Animations</title>
  <style>
    @keyframes fade {
      0%, 100% {
        opacity: 0;
      }
      50% {
        opacity: 1;
      }
    }
    .animated-box {
      width: 200px;
      height: 200px;
      background-color: #333;
      animation: fade 3s infinite;
    }
  </style>
~~~

```
</head>

<body>

  <div class="animated-box"></div>

</body>

</html>
```

Scenario 6: Responsive Image Gallery

Task: Create a responsive image gallery with CSS Grid that adjusts the number of columns based on screen size.

Solution:

```html
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Responsive Image Gallery</title>

  <style>

    .gallery {

      display: grid;

      grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));

      gap: 10px;

    }

    .gallery img {

      width: 100%;

      height: auto;

    }

  </style>
```

```
</head>
<body>
  <div class="gallery">
    <img src="https://via.placeholder.com/200" alt="Image 1">
    <img src="https://via.placeholder.com/200" alt="Image 2">
    <img src="https://via.placeholder.com/200" alt="Image 3">
    <img src="https://via.placeholder.com/200" alt="Image 4">
  </div>
</body>
</html>
```

Scenario 7: Custom Checkbox and Radio Buttons
Task: Style checkbox and radio buttons with custom CSS.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Custom Checkbox and Radio Buttons</title>
  <style>
    input[type="checkbox"], input[type="radio"] {
      display: none;
    }
    label {
      cursor: pointer;
    }
```

```
        .custom-checkbox, .custom-radio {
            display: inline-block;
            width: 20px;
            height: 20px;
            border: 2px solid #333;
            border-radius: 4px;
            position: relative;
        }
        .custom-radio {
            border-radius: 50%;
        }
        input[type="checkbox"]:checked + .custom-checkbox::after,
input[type="radio"]:checked + .custom-radio::after {
            content: '';
            position: absolute;
            top: 50%;
            left: 50%;
            width: 12px;
            height: 12px;
            background-color: #333;
            transform: translate(-50%, -50%);
        }
        .custom-radio::after {
            border-radius: 50%;
        }
    </style>
</head>
<body>
    <label>
        <input type="checkbox">
        <div class="custom-checkbox"></div> Checkbox
    </label>
    <label>
        <input type="radio" name="radio">
        <div class="custom-radio"></div> Radio
    </label>
</body>
</html>
```

Scenario 8: Sticky Header
Task: Create a sticky header that remains at the top of the page while scrolling.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sticky Header</title>
    <style>
        body {
            margin: 0;
            font-family: Arial, sans-serif;
        }
        .header {
            background-color: #333;
            color: white;
            padding: 10px;
```

```
      position: sticky;
      top: 0;
      z-index: 1000;
    }
    .content {
      height: 2000px;

      padding: 20px;
    }
  </style>
</head>
<body>
  <div class="header">Sticky Header</div>
  <div class="content">
    Scroll down to see the sticky header in action.
  </div>
</body>
</html>
```

Scenario 9: Custom Scrollbar
Task: Style the scrollbar with custom colors and width.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Custom Scrollbar</title>
  <style>
    body {
      margin: 0;
      height: 200vh;
      background-color: #f0f0f0;
    }
    ::-webkit-scrollbar {
      width: 12px;
    }
    ::-webkit-scrollbar-track {
      background: #f0f0f0;
    }
    ::-webkit-scrollbar-thumb {
      background: #333;
      border-radius: 6px;
    }
    ::-webkit-scrollbar-thumb:hover {
      background: #555;
    }
  </style>
</head>
<body>
  Scroll to see the custom scrollbar.
</body>
</html>
```

Scenario 10: Text Shadow Effect

Task: Apply a text shadow effect to create a 3D look.

Solution:
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Text Shadow Effect</title>
    <style>
        .text {
            font-size: 48px;
            font-weight: bold;
            color: #333;
            text-shadow: 2px 2px 5px rgba(0, 0, 0, 0.3);
        }
    </style>
</head>
<body>
    <div class="text">3D Text Shadow</div>
</body>
</html>
```