HTML/CSS Cool Snippets for Your Website

Mrs. Nada M. Salem

The following effects will definitely make your Google Site stand out with unique visuals and interactivity that go beyond standard website features!

1. Cancelled

---

2. Create a Clickable Button (Beginner Level)

This code creates a clickable button that links to another website. Customize the button's text and where it links to.

```
<a href="https://example.com" target="_blank">
    <button style="padding: 10px 20px; font-size: 16px;
background-color: #4CAF50; color: white; border: none;
border-radius: 5px;">Visit My Favorite Site</button>
</a>
```

Replace the URL in the `href` with the site you want to link to, and change the button text.

---

3. Add Scrolling Text (Marquee) (Beginner Level)

This code will create text that scrolls across the screen. You can use it to make announcements or emphasize key points.

```
<marquee behavior="scroll" direction="left">Welcome to My Cool
Website!</marquee>
```

You can change the direction and text as needed.

---

4. Typewriter Text Animation (Intermediate Level)

Create an animated typewriter effect for your headings or important text. It makes your website more interactive.

```
<style>
  .typewriter h1 {
      overflow: hidden;
      border-right: .15em solid orange;
      white-space: nowrap;
      margin: 0 auto;
      letter-spacing: .15em;
      animation: typing 3.5s steps(30, end), blink-caret .75s
step-end infinite;
  }

  @keyframes typing {
      from { width: 0 }
      to { width: 100% }
  }

  @keyframes blink-caret {
      from, to { border-color: transparent }
      50% { border-color: orange; }
  }
</style>

<div class="typewriter">
  <h1>Welcome to My Awesome Website!</h1>
</div>
```

---

5. Create a Parallax Scrolling Effect (Intermediate Level)

Parallax scrolling gives your website a 3D effect where the background moves at a different speed than the foreground.

```
<style>
```

```
  .parallax {
      background-image: url('https://example.com/yourimage.jpg');
      height: 500px;
      background-attachment: fixed;
      background-position: center;
      background-repeat: no-repeat;
      background-size: cover;
  }
</style>

<div class="parallax"></div>
<div style="height: 400px; background-color: #f4f4f4;">
  <h2>Scroll down to see the parallax effect!</h2>
</div>
```

Replace yourimage.jpg with your own image URL.

---

6. Add an Image Gallery with Hover Effect (Intermediate Level)

This gallery displays images, and when you hover over them, they zoom in slightly.

```
<style>
  .gallery img {
      width: 100px;
      height: 100px;
      transition: transform 0.3s ease;
  }
  .gallery img:hover {
      transform: scale(1.2);
  }
</style>

<div class="gallery">
    <img src="https://example.com/image1.jpg" alt="Image 1">
    <img src="https://example.com/image2.jpg" alt="Image 2">
    <img src="https://example.com/image3.jpg" alt="Image 3">
```

```
</div>
```

Replace the image URLs with your own pictures.

---

7. Glowing Button (Advanced Level)

Make your buttons glow with this cool hover effect.

```
<style>
  .glow-button {
      font-size: 16px;
      padding: 10px 20px;
      color: white;
      background-color: #ff4081;
      border: none;
      border-radius: 5px;
      box-shadow: 0 0 20px rgba(255, 64, 129, 0.5);
      transition: all 0.4s ease-in-out;
  }

  .glow-button:hover {
      box-shadow: 0 0 50px rgba(255, 64, 129, 1);
      transform: scale(1.1);
  }
</style>

<button class="glow-button">Click Me!</button>
```

---

8. Floating Animation (Advanced Level)

Use this effect to make objects like images float up and down on your page.

```
<style>
```

```
  .float-animation {
      position: relative;
      animation: float 4s ease-in-out infinite;
  }

  @keyframes float {
      0% { transform: translateY(0px); }
      50% { transform: translateY(-20px); }
      100% { transform: translateY(0px); }
  }
</style>

<img class="float-animation" src="https://example.com/yourimage.jpg"
alt="Floating Image" width="200px">
```

---

9. Particle Background (Advanced Level)

Add a dynamic particle background to your page for a futuristic look.

```
<style>
  .particle-background {
      position: relative;
      width: 100%;
      height: 100vh;
      background-color: #000;
      overflow: hidden;
  }

  .particle {
      position: absolute;
      width: 6px;
      height: 6px;
      background-color: rgba(255, 255, 255, 0.6);
      border-radius: 50%;
```

```
        animation: particle-move 10s infinite linear;
    }

    @keyframes particle-move {
        from {
            transform: translateY(0) translateX(0);
        }
        to {
            transform: translateY(-100vh) translateX(100vw);
        }
    }
</style>

<div class="particle-background">
    <div class="particle" style="top: 90%; left: 20%;
animation-duration: 6s;"></div>
    <div class="particle" style="top: 95%; left: 50%;
animation-duration: 8s;"></div>
    <div class="particle" style="top: 85%; left: 70%;
animation-duration: 7s;"></div>
</div>
```

---

10. Neon Glow Text Effect (Advanced Level)

This code creates glowing neon text that pulsates.

```
<style>
    .neon-text {
        color: #fff;
        text-shadow: 0 0 5px #0ff, 0 0 10px #0ff, 0 0 20px #0ff, 0 0
40px #0ff;
        font-size: 48px;
        font-family: 'Arial', sans-serif;
        letter-spacing: 2px;
```

```
      animation: glow 1.5s ease-in-out infinite alternate;
  }

  @keyframes glow {
      from {
          text-shadow: 0 0 5px #0ff, 0 0 10px #0ff, 0 0 20px #0ff, 0
0 40px #0ff;
      }
      to {
          text-shadow: 0 0 20px #ff00ff, 0 0 30px #ff00ff, 0 0 50px
#ff00ff;
      }
  }
</style>

<h1 class="neon-text">Cool Neon Text!</h1>
```

---

11. Particles Background (CSS Only)(Advanced Level)

Create a particle effect background where small circles move randomly, mimicking a stars or bubbles effect.

```
<style>
.particle-background {
    position: relative;
    width: 100%;
    height: 100vh;
    background-color: #000;
    overflow: hidden;
}

.particle {
    position: absolute;
    width: 6px;
    height: 6px;
```

```
    background-color: rgba(255, 255, 255, 0.6);
    border-radius: 50%;
    animation: particle-move 10s infinite linear;
}

@keyframes particle-move {
    from {
        transform: translateY(0) translateX(0);
    }
    to {
        transform: translateY(-100vh) translateX(100vw);
    }
}
</style>

<div class="particle-background">
    <div class="particle" style="top: 90%; left: 20%;
animation-duration: 6s;"></div>
    <div class="particle" style="top: 95%; left: 50%;
animation-duration: 8s;"></div>
    <div class="particle" style="top: 85%; left: 70%;
animation-duration: 7s;"></div>
    <!-- Add more particles as desired -->
</div>
```

This creates a cool particles animation, making it look like floating bubbles or stars in space. (On Google Sites, add it as an animation only, not as a background)

---

12. CSS Text Stroke (Outlined Text) (Advanced Level)

Create cool outlined text that makes your titles pop.

```
<style>
.stroke-text {
```

```
    font-size: 72px;
    color: transparent;
    -webkit-text-stroke-width: 2px;
    -webkit-text-stroke-color: #ff0000;
}
</style>

<h1 class="stroke-text">Outlined Text</h1>
```

This code gives a stroke or outline to the text, which can be customized by changing the stroke color or thickness.

---

13. Particle Hover Effect on Buttons (Advanced Level)

Create a cool button with particles that move around when hovered.

```
<style>
.particle-button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 15px 30px;
    font-size: 16px;
    position: relative;
    overflow: hidden;
    transition: background-color 0.4s ease;
}

.particle-button:hover {
    background-color: #2980b9;
}

.particle-button::before, .particle-button::after {
    content: '';
```

```
    position: absolute;
    top: -5px;
    width: 100%;
    height: 5px;
    background: radial-gradient(circle, #fff, transparent);
    animation: move 2s linear infinite;
}

.particle-button::after {
    top: auto;
    bottom: -5px;
}

@keyframes move {
    from { left: -100%; }
    to { left: 100%; }
}
</style>

<button class="particle-button">Hover me</button>
```

This button will show particle-like effects on hover. It gives a more engaging interaction to users.

---

14. CSS Gradient Border Animation (Advanced Level)

Add a gradient border that animates around an element, making it stand out.

```
<style>
.gradient-border {
    padding: 20px;
    background-color: white;
    position: relative;
    border: 2px solid transparent;
```

```
    border-radius: 5px;
    background-clip: padding-box;
}

.gradient-border:before {
    content: '';
    position: absolute;
    top: -2px; left: -2px; right: -2px; bottom: -2px;
    background: linear-gradient(45deg, #ff0000, #00ff00, #0000ff,
#ff00ff);
    background-size: 400% 400%;
    z-index: -1;
    border-radius: 5px;
    animation: gradient-border 6s ease infinite;
}

@keyframes gradient-border {
    0% { background-position: 0% 50%; }
    50% { background-position: 100% 50%; }
    100% { background-position: 0% 50%; }
}
</style>

<div class="gradient-border">
    <h2>Cool Animated Border</h2>
    <p>This box has an animated gradient border!</p>
</div>
```

This creates an eye-catching gradient border around content, making it stand out with a nice, continuous animation.

---

15. CSS Speech Bubble (Advanced Level)

Create speech bubbles for images or avatars; perfect for personalizing content.

```
<style>
```

```
.speech-bubble {
    position: relative;
    background: #00aabb;
    border-radius: .4em;
    padding: 20px;
    color: white;
    width: 250px;
}

.speech-bubble::after {
    content: '';
    position: absolute;
    bottom: 100%;
    left: 50px;
    border-width: 0 10px 10px 10px;
    border-style: solid;
    border-color: transparent transparent #00aabb transparent;
}
</style>

<div class="speech-bubble">
    <p>Hi there! I am a cool speech bubble.</p>
</div>
```

This code creates a speech bubble effect that can be customized and positioned near an avatar or profile image.

---

16. CSS Firefly Animation (Moving Glowing Dots) (Advanced Level)

This creates a "firefly" effect with moving, glowing dots across the background, which adds a magical atmosphere to the website.

```
<style>
.fireflies {
    position: absolute;
    width: 100%;
```

```
    height: 100vh;
    background-color: #000;
    overflow: hidden;
}

.firefly {
    position: absolute;
    width: 5px;
    height: 5px;
    background-color: rgba(255, 255, 0, 0.8);
    border-radius: 50%;
    animation: fly 10s infinite linear;
    box-shadow: 0 0 10px rgba(255, 255, 0, 0.8), 0 0 20px rgba(255,
255, 0, 0.8);
}

@keyframes fly {
    from {
        transform: translateX(0) translateY(0);
    }
    to {
        transform: translateX(calc(100vw - 50px))
translateY(calc(100vh - 50px));
    }
}
</style>

<div class="fireflies">
    <div class="firefly" style="top: 20%; left: 30%;"></div>
    <div class="firefly" style="top: 40%; left: 70%;"></div>
    <div class="firefly" style="top: 80%; left: 50%;"></div>
    <!-- Add more fireflies as desired -->
</div>
```

This will add small, glowing firefly-like dots floating around the background.

---

17. CSS 3D Text Effect (Advanced Level)

Make text pop out from the page with a 3D effect.

```
<style>
.three-d-text {
    font-size: 72px;
    font-weight: bold;
    color: #fff;
    position: relative;
    text-transform: uppercase;
}

.three-d-text:before {
    content: attr(data-text);
    position: absolute;
    top: 0;
    left: 5px;
    color: #ff0000;
    z-index: -1;
    transform: skew(10deg, 10deg);
}

.three-d-text:after {
    content: attr(data-text);
    position: absolute;
    top: 5px;
    left: 0;
    color: #0000ff;
    z-index: -1;
    transform: skew(10deg, 10deg);
}
</style>

<h1 class="three-d-text" data-text="3D Text!">3D Text!</h1>
```

This code gives your text a cool retro-style 3D look with red and blue shadow layers.

---

18. CSS Ripple Effect on Buttons (Advanced Level)

Create a beautiful ripple effect that spreads across the button when clicked.

```
<style>
.ripple-button {
    position: relative;
    overflow: hidden;
    padding: 10px 20px;
    font-size: 18px;
    color: #fff;
    background-color: #3498db;
    border: none;
    border-radius: 4px;
    outline: none;
}

.ripple-button::after {
    content: "";
    position: absolute;
    width: 100px;
    height: 100px;
    background: rgba(255, 255, 255, 0.4);
    display: block;
    border-radius: 50%;
    transform: scale(0);
    opacity: 1;
    animation: ripple 0.6s ease-out;
    pointer-events: none;
}

@keyframes ripple {
    to {
```

```
        transform: scale(2);
        opacity: 0;
    }
}
</style>

<button class="ripple-button">Click Me</button>
```

When clicked, this button will have a smooth ripple animation that spreads out across its surface. We will be able to see the ripple effect only once. **Can you make it loop??**

---

19. CSS Glitch Effect (Advanced Level)

Add a glitchy, distorted animation to your headings for a modern and techy look.

```
<style>
.glitch {
    font-size: 72px;
    color: white;
    position: relative;
}

.glitch::before, .glitch::after {
    content: attr(data-text);
    position: absolute;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background: black;
    overflow: hidden;
}
```

```
.glitch::before {
    left: 2px;
    text-shadow: -2px 0 red;
    clip: rect(0, 140px, 50px, 0);
    animation: glitch 2s infinite;
}

.glitch::after {
    left: -2px;
    text-shadow: -2px 0 blue;
    clip: rect(85px, 9999px, 140px, 0);
    animation: glitch 3s infinite;
}

@keyframes glitch {
    0% { clip: rect(42px, 9999px, 44px, 0); }
    5% { clip: rect(85px, 9999px, 92px, 0); }
    10% { clip: rect(24px, 9999px, 26px, 0); }
    15% { clip: rect(55px, 9999px, 60px, 0); }
    20% { clip: rect(12px, 9999px, 14px, 0); }
    25% { clip: rect(33px, 9999px, 38px, 0); }
    30% { clip: rect(75px, 9999px, 78px, 0); }
    35% { clip: rect(0px, 9999px, 5px, 0); }
    40% { clip: rect(120px, 9999px, 122px, 0); }
    100% { clip: rect(85px, 9999px, 87px, 0); }
}
</style>

<h1 class="glitch" data-text="Glitch Effect!">Glitch Effect!</h1>
```

This will make the text appear glitchy, which is perfect for futuristic or tech-themed websites.

---

20. CSS Animated Progress Bars (Advanced Level)

Create animated progress bars that could be used to display skills, goals, or project statuses.

```
<style>
.progress-bar {
    background-color: #e0e0e0;
    border-radius: 25px;
    padding: 5px;
    width: 300px;
    margin: 20px 0;
}

.progress-bar-fill {
    height: 20px;
    border-radius: 20px;
    background-color: #76c7c0;
    width: 0%;
    animation: fill 2s forwards;
}

@keyframes fill {
    to {
        width: 80%;
    }
}
</style>

<div class="progress-bar">
    <div class="progress-bar-fill"></div>
</div>
```

The fill animation can be customized to different percentages by changing the width in the `@keyframes` rule.

---

21. Dynamic Countdown Timer (Advanced Level)

Create a real-time countdown timer that counts down to a specific event.

```
<script>
  function countdown() {
    var countDownDate = new Date("March 7, 2025
23:59:59").getTime();
    var x = setInterval(function() {
      var now = new Date().getTime();
      var distance = countDownDate - now;
      var days = Math.floor(distance / (1000 * 60 * 60 * 24));
      var hours = Math.floor((distance % (1000 * 60 * 60 * 24)) /
(1000 * 60 * 60));
      var minutes = Math.floor((distance % (1000 * 60 * 60)) / (1000
* 60));
      var seconds = Math.floor((distance % (1000 * 60)) / 1000);
      document.getElementById("timer").innerHTML = days + "d " +
hours + "h " + minutes + "m " + seconds + "s ";
      if (distance < 0) {
        clearInterval(x);
        document.getElementById("timer").innerHTML = "EXPIRED";
      }
    }, 1000);
  }
  window.onload = countdown;
</script>

<h2>Event Countdown:</h2><p id="timer"></p>
```

---

22. CSS Grid Masonry Layout

Utilize CSS Grid to create a masonry-style layout, allowing for responsive, Pinterest-like arrangements without relying on JavaScript.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Grid Masonry Layout</title>
  <style>
    .masonry {
      display: grid;
      grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
      grid-auto-rows: 10px;
      gap: 10px;
    }
    .masonry-item {
      background-color: #8e44ad;
      border-radius: 5px;
      padding: 10px;
      color: white;
      font-size: 1.2em;
    }
    .masonry-item-content {
      grid-row-end: span 30;
    }
  </style>
</head>
<body>
  <div class="masonry">
    <div class="masonry-item masonry-item-content">Item 1</div>
    <div class="masonry-item masonry-item-content">Item 2</div>
    <div class="masonry-item masonry-item-content">Item 3</div>
    <!-- Add more items as needed -->
  </div>
</body>
</html>
```

---

23. CSS Clip-Path Hover Effects

Apply the `clip-path` property to images to create dynamic hover effects, enhancing visual engagement on your website.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Clip-Path Hover Effects</title>
  <style>
    .clip-hover {
      width: 300px;
      height: 200px;
      overflow: hidden;
      position: relative;
    }
    .clip-hover img {
      width: 100%;
      height: 100%;
      transition: transform 0.5s ease-in-out;
    }
    .clip-hover:hover img {
      transform: scale(1.1);
      clip-path: polygon(0 0, 100% 0, 100% 100%, 0 100%);
    }
  </style>
</head>
<body>
  <div class="clip-hover">
    <img src="https://via.placeholder.com/300x200" alt="Sample Image">
  </div>
</body>
</html>
```

24. CSS Variables for Theming

Implement CSS variables to establish a theming system, enabling consistent and maintainable design across your website.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Variables for Theming</title>
  <style>
    :root {
      --primary-color: #3498db;
      --secondary-color: #2ecc71;
      --font-color: #ffffff;
    }
    body {
      background-color: var(--primary-color);
      color: var(--font-color);
      font-family: Arial, sans-serif;
      text-align: center;
      padding: 50px;
    }
    .button {
      background-color: var(--secondary-color);
      border: none;
      color: var(--font-color);
      padding: 15px 32px;
      text-align: center;
      text-decoration: none;
      display: inline-block;
      font-size: 16px;
      margin: 4px 2px;
      cursor: pointer;
      border-radius: 4px;
    }
  </style>
</head>
<body>
  <h1>Welcome to My Themed Website</h1>
```

```
  <button class="button">Click Me</button>
</body>
</html>
```

25. CSS Scroll Snap

Use CSS Scroll Snap to create smooth, snap-to-position scrolling experiences, improving navigation through content sections.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Scroll Snap</title>
  <style>
    html, body {
      height: 100%;
      margin: 0;
      scroll-behavior: smooth;
    }
    .container {
      height: 100%;
      scroll-snap-type: y mandatory;
      overflow-y: scroll;
    }
    .section {
      height: 100vh;
      scroll-snap-align: start;
      display: flex;
      justify-content: center;
      align-items: center;
      font-size: 2em;
      color: white;
    }
    .section:nth-child(odd) {
      background-color: #1abc9c;
```

```
    }
    .section:nth-child(even) {
      background-color: #e74c3c;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="section">Section 1</div>
    <div class="section">Section 2</div>
    <div class="section">Section 3</div>
    <!-- Add more sections as needed -->
  </div>
</body>
</html>
```

26. Responsive CSS Grid Layout

Utilize CSS Grid to create a responsive layout that adjusts to various screen sizes.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Responsive CSS Grid Layout</title>
  <style>
    .grid-container {
      display: grid;
      grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
      gap: 10px;
    }
    .grid-item {
      background-color: #76c7c0;
      padding: 20px;
      text-align: center;
      color: white;
      font-size: 1.2em;
```

```
    }
  </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 class="grid-item">Item 4</div>
    <div class="grid-item">Item 5</div>
    <div class="grid-item">Item 6</div>
  </div>
</body>
</html>
```

27. Flexbox Vertical Alignment

Center content both vertically and horizontally using Flexbox.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Flexbox Vertical Alignment</title>
  <style>
    .flex-container {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background-color: #f4f4f4;
    }
    .flex-item {
      background-color: #3498db;
      padding: 20px;
      color: white;
      font-size: 1.5em;
    }
```

```
  </style>
</head>
<body>
  <div class="flex-container">
    <div class="flex-item">Centered Content</div>
  </div>
</body>
</html>
```

28. SVG Icon Integration

Incorporate scalable vector graphics (SVG) for crisp and resolution-independent icons.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>SVG Icon Integration</title>
  <style>
    .icon {
      width: 50px;
      height: 50px;
    }
  </style>
</head>
<body>
  <svg class="icon" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
    <path d="M12 2L2 7l10 5 10-5-10-5zm0 11l-10-5v10l10 5 10-5V8l-10 5z"
fill="#000"/>
  </svg>
</body>
</html>
```

29. CSS Shape Outside for Text Wrapping

Use the shape-outside property to wrap text around custom shapes.

This property enables text to wrap around non-rectangular float elements, allowing for creative layouts.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Shape Outside for Text Wrapping</title>
  <style>
    .float-image {
      float: left;
      shape-outside: circle(50%);
      width: 200px;
      height: 200px;
      margin: 20px;
      background: url('https://via.placeholder.com/200') no-repeat
center/cover;
      clip-path: circle(50%);
    }
    .text {
      font-size: 1.2em;
      line-height: 1.6;
    }
  </style>
</head>
<body>
  <div class="float-image"></div>
  <div class="text">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque sit
amet accumsan tortor. Nulla facilisi. Donec vel libero at lectus rutrum
vestibulum vitae ut turpis. Ut ultricies pulvinar turpis, nec convallis
nunc. Nullam ac sapien sit amet justo malesuada dapibus. Integer vel
turpis ac quam condimentum bibendum. Aliquam erat volutpat. Suspendisse
potenti. Sed nec lorem vel orci varius congue.
  </div>
</body>
</html>
```

30. CSS Animated Navigation Menu

Design a navigation menu with smooth hover animations, providing a dynamic user experience.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Animated Navigation Menu</title>
  <style>
    .nav-menu {
      display: flex;
      list-style: none;
      background-color: #333;
      padding: 0;
    }
    .nav-item {
      flex: 1;
    }
    .nav-link {
      display: block;
      padding: 15px;
      color: white;
      text-align: center;
      text-decoration: none;
      transition: background-color 0.3s ease;
    }
    .nav-link:hover {
      background-color: #555;
    }
  </style>
</head>
<body>
  <ul class="nav-menu">
    <li class="nav-item"><a href="#" class="nav-link">Home</a></li>
    <li class="nav-item"><a href="#" class="nav-link">About</a></li>
    <li class="nav-item"><a href="#" class="nav-link">Services</a></li>
    <li class="nav-item"><a href="#" class="nav-link">Contact</a></li>
  </ul>
</body>
```

```
</html>
```

31. CSS Card Flip Animation

Implement a card that flips on hover, revealing additional information on the back side, ideal for showcasing content interactively.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Card Flip Animation</title>
  <style>
    .card {
      perspective: 1000px;
      width: 200px;
      height: 300px;
    }
    .card-inner {
      position: relative;
      width: 100%;
      height: 100%;
      transition: transform 0.6s;
      transform-style: preserve-3d;
    }
    .card:hover .card-inner {
      transform: rotateY(180deg);
    }
    .card-front,
    .card-back {
      position: absolute;
      width: 100%;
      height: 100%;
      backface-visibility: hidden;
      display: flex;
      justify-content: center;
      align-items: center;
```

```
      font-size: 1.5em;
      color: white;
    }
    .card-front {
      background-color: #333;
    }
    .card-back {
      background-color: #555;
      transform: rotateY(180deg);
    }
  </style>
</head>
<body>
  <div class="card">
    <div class="card-inner">
      <div class="card-front">Front</div>
      <div class="card-back">Back</div>
    </div>
  </div>
</body>
</html>
```

32. CSS Sticky Footer

Create a footer that sticks to the bottom of the page, ensuring it remains in place regardless of the content length.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sticky Footer</title>
  <style>
    body {
      display: flex;
      flex-direction: column;
      min-height: 100vh;
```

```
      margin: 0;
    }
    .content {
      flex: 1;
    }
    .footer {
      background-color: #333;
      color: white;
      text-align: center;
      padding: 10px;
    }
  </style>
</head>
<body>
  <div class="content">
    <!-- Main content goes here -->
  </div>
  <div class="footer">
    Sticky Footer Content
  </div>
</body>
</html>
```

33. CSS Responsive Accordion

Create a responsive accordion component that expands and collapses sections of content, enhancing content organization and user interaction.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Responsive Accordion</title>
  <style>
    .accordion {
      width: 100%;
```

```
      max-width: 600px;
      margin: 20px auto;
    }
    .accordion-item {
      border-bottom: 1px solid #ddd;
    }
    .accordion-header {
      background-color: #f4f4f4;
      padding: 15px;
      cursor: pointer;
      transition: background-color 0.3s ease;
    }
    .accordion-header:hover {
      background-color: #e2e2e2;
    }
    .accordion-content {
      max-height: 0;
      overflow: hidden;
      transition: max-height 0.3s ease;
      padding: 0 15px;
    }
    .accordion-content p {
      margin: 15px 0;
    }
    .accordion-item.active .accordion-content {
      max-height: 200px;
    }
  </style>
</head>
<body>
  <div class="accordion">
    <div class="accordion-item">
      <div class="accordion-header">Section 1</div>
      <div class="accordion-content">
        <p>Content for section 1.</p>
      </div>
    </div>
    <div class="accordion-item">
      <div class="accordion-header">Section 2</div>
      <div class="accordion-content">
```

```
        <p>Content for section 2.</p>
      </div>
    </div>
    <div class="accordion-item">
      <div class="accordion-header">Section 3</div>
      <div class="accordion-content">
        <p>Content for section 3.</p>
      </div>
    </div>
  </div>

  <script>
    const items = document.querySelectorAll('.accordion-header');
    items.forEach(item => {
      item.addEventListener('click', () => {
        const parent = item.parentElement;
        parent.classList.toggle('active');
        const content = item.nextElementSibling
::contentReference[oaicite:8]{index=8}
```

34. CSS 3D Cube Animation

Create a rotating 3D cube using CSS animations and transforms, providing an interactive visual effect.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>3D Cube Animation</title>
  <style>
    .scene {
      width: 200px;
      height: 200px;
      perspective: 600px;
    }
    .cube {
```

```
      width: 100%;
      height: 100%;
      position: relative;
      transform-style: preserve-3d;
      transform: rotateX(-30deg) rotateY(-45deg);
      animation: rotate 5s infinite linear;
    }
    .face {
      position: absolute;
      width: 200px;
      height: 200px;
      background: rgba(255, 255, 255, 0.9);
      border: 2px solid #ccc;
    }
    .front  { transform: translateZ(100px); }
    .back   { transform: rotateY(180deg) translateZ(100px); }
    .right  { transform: rotateY(90deg) translateZ(100px); }
    .left   { transform: rotateY(-90deg) translateZ(100px); }
    .top    { transform: rotateX(90deg) translateZ(100px); }
    .bottom { transform: rotateX(-90deg) translateZ(100px); }
    @keyframes rotate {
      from { transform: rotateX(-30deg) rotateY(-45deg); }
      to { transform: rotateX(-30deg) rotateY(315deg); }
    }
  </style>
</head>
<body>
  <div class="scene">
    <div class="cube">
      <div class="face front">Front</div>
      <div class="face back">Back</div>
      <div class="face right">Right</div>
      <div class="face left">Left</div>
      <div class="face top">Top</div>
      <div class="face bottom">Bottom</div>
    </div>
  </div>
</body>
</html>
```

These advanced techniques can enhance the functionality and aesthetics of your web projects, providing modern solutions to common design challenges.