Internship Report – Frontend Dev(Week3)

Name: Zainab
Father Name: Assad Qayyum
Date: 9th July, 2025
Internship Domain: Front-end Intern
Task: CSS - Transitions and Animations, Pseudo-classes & Pseudo-elements

Task Overview: (Day3)

The purpose of this task was to explore and practice advanced CSS concepts that enhance user interactivity and visual appeal: **Transitions, Animations, Pseudo-classes,** and **Pseudo-elements**.

Content Covered:

- CSS Transitions and their properties
- CSS Animations and their properties
- CSS Pseudo-classes
- CSS Pseudo-elements

Introduction:

In modern web design, creating interactive and visually appealing interfaces is essential. CSS have tools like transitions and animations to make style changes feel smooth and engaging. Pseudo-classes and pseudo-elements further enhance styling by allowing specific interactions and targeting parts of elements.

1. CSS Transitions:

Transitions let you define how a CSS property changes over time — instead of changing instantly, the change happens smoothly.

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Syntax:

```
selector {
  transition: [property] [duration] [timing-function] [delay];
}
```

Transition properties include; Color, background-color, width, height, opacity, transform

Common Transition Properties:

Property	Description
transition-property	Specifies the CSS property the transition effect is for (color, background-color)
transition-duration	How many sec/msec the transition takes to complete (e.g., 0.5s, 1s)
transition-timing-function	Defines speed curve of transition effect(ease, linear, etc.)
transition-delay	Delay before the transition starts

Example: Smoothly changes the background color when hovered.

```
.box {
  transition: background-color 0.3s ease;
}
.box:hover {
  background-color: green;
}
```

2. CSS Animations:

CSS Animations allow defining keyframes to change CSS styles over time, allowing looping and more complex movements than transitions.

- An animation lets an element gradually change from one style to another.
- You can change as many CSS properties as you want.
- To use CSS animation, you must first specify some keyframes.
- Keyframes hold what styles the element will have at certain times.

Syntax:

```
@keyframes animationName {
  0%   { property: value1; }
  50%  { property: value2; }
  100% { property: value3; }
}
.element {
  animation: animationName duration timing-function delay iteration-count direction;
}
```

Common Animation Properties:

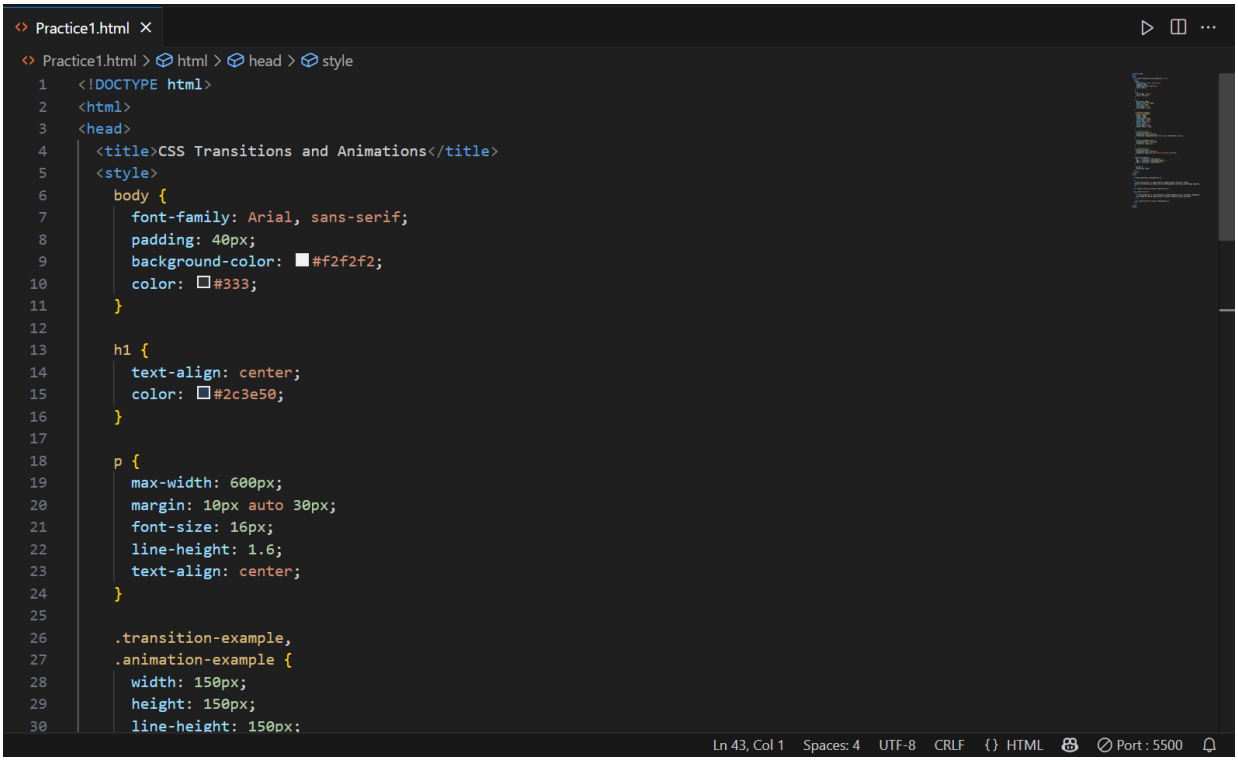
Property	Description
@keyframes	Defines the animation steps (start, middle, end)
animation-name	The name linked to the @keyframes
animation-duration	How long each cycle of animation lasts
animation-timing-function	Speed curve (ease, ease-in-out, linear, etc.)
animation-delay	Delay before animation starts
animation-iteration-count	How many times to repeat the animation (infinite, 1, etc.)
animation-direction	Direction (normal, reverse, alternate, etc.)

Example: Moves the box back and forth continuously.

```
@keyframes slide {
  from { transform: translateX(0); }
  to { transform: translateX(100px); }
}
.box {
  animation: slide 2s infinite alternate;
}
```

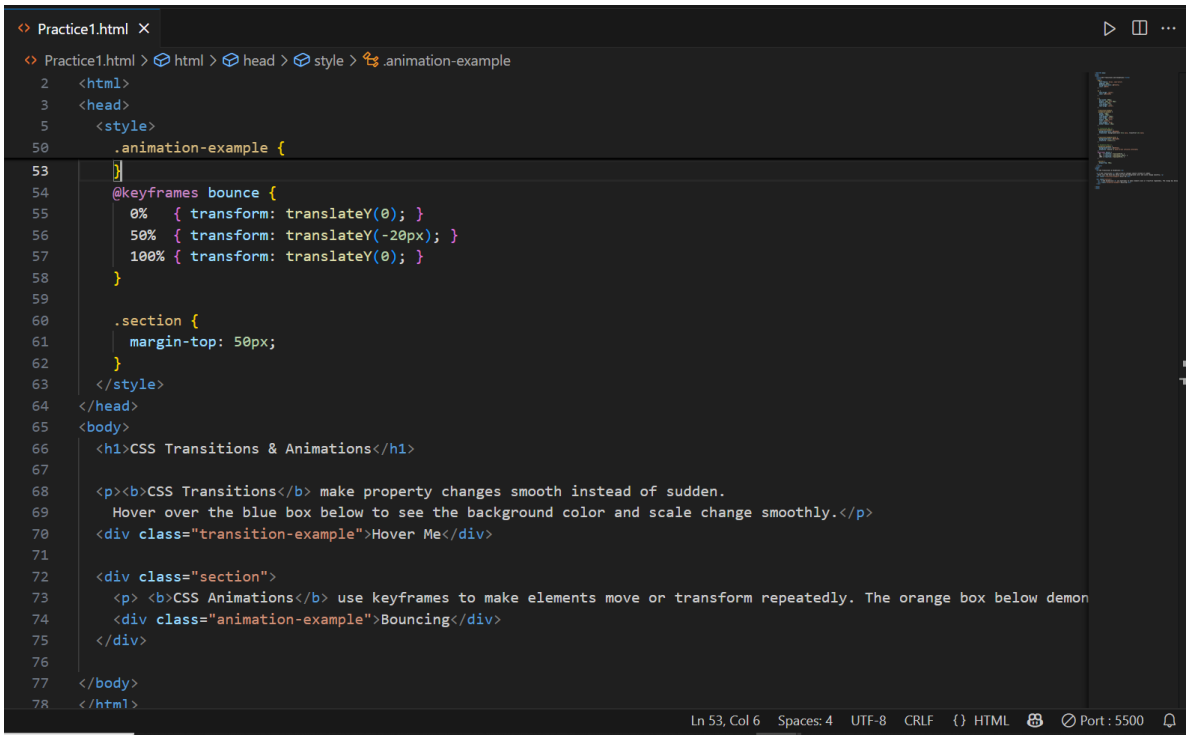
Practice Code: (Transition and Animation)

The code below demonstrates both CSS transitions (on hover) and CSS animations (continuous movement) with clear, visible effects.



```
Practice1.html
Practice1.html > html > head > style
<!DOCTYPE html>
<html>
<head>
  <title>CSS Transitions and Animations</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      padding: 40px;
      background-color: #f2f2f2;
      color: #333;
    }

    h1 {
      text-align: center;
      color: #2c3e50;
    }

    p {
      max-width: 600px;
      margin: 10px auto 30px;
      font-size: 16px;
      line-height: 1.6;
      text-align: center;
    }

    .transition-example,
    .animation-example {
      width: 150px;
      height: 150px;
      line-height: 150px;
Ln 43, Col 1  Spaces: 4  UTF-8  CRLF  { } HTML  Port : 5500
```

```
Practice1.html
Practice1.html > html > head > style
<html>
<head>
  <style>
    .transition-example,
    .animation-example {
      height: 150px;
      line-height: 150px;
      text-align: center;
      margin: 30px auto;
      color: white;
      font-weight: bold;
      border-radius: 10px;
    }

    /* Transition Box */
    .transition-example {
      background-color: #3498db;
      transition: background-color 0.4s ease, transform 0.3s ease;
    }

    .transition-example:hover {
      background-color: #27ae60;
      transform: scale(1.1);
    }

    /* Animation Box */
    .animation-example {
      background-color: #e67e22;
      animation: bounce 2s ease-in-out infinite alternate;
    }
Ln 43, Col 1  Spaces: 4  UTF-8  CRLF  { } HTML  Port : 5500
```



3. CSS Pseudo-classes:

Pseudo-classes target an element in a specific state or condition (like hover, focus, first child). A pseudo-class is used to define a special state of an element.

For example, it can be used to:

- Style an element when a user moves the mouse over it
- Style visited and unvisited links differently
- Style an element when it gets focus
- Style valid/invalid/required/optional form elements

Syntax:

```
selector:pseudo-class {
  property: value;
}
```

Common Pseudo-classes:

Pseudo-class	**Description**
:hover	Applies when mouse hovers on an element
:focus	Applies when input is focused
:active	When element is clicked
:checked	Applies to checked radio/checkbox
:nth-child(n)	Targets specific children
:first-child	Targets first child of a parent

Example:

```
/* Hover state for buttons */
button:hover {
  background-color: green;
}

/* Input field when focused */
input:focus {
  border-color: blue;
}

/* First item in a list */
li:first-child {
  font-weight: bold;
}
```

4. CSS Pseudo-elements:

Pseudo-elements let you style parts of an element (like the first letter) or insert content using CSS. A CSS pseudo-element is used to style specific parts of an element.

For example, it can be used to:

- Style the first letter or line, of an element
- Insert content before or after an element
- Style the markers of list items
- Style the viewbox behind a dialog box

Syntax:

```
selector::pseudo-element {
  property: value;
}
```

Common Pseudo-elements:

Pseudo-element	Description
::before	Inserts content before the element
::after	Inserts content after the element
::first-letter	Styles the first letter of a block of text
::first-line	Styles the first line of a block of text

Example: Styles the first letter of a paragraph and adds an icon before a heading.

```
p::first-letter {
  font-size: 24px;
  color: red;
}

h2::before {
  content: "★ ";
  color: gold;
}
```

Practice Code: (Pseudo-classes and Pseudo-elements)

This code demonstrates CSS **pseudo-classes** like **:hover, :focus,** and **:first-child**, and **pseudo-elements** like **::first-letter** and **::first-line** to style specific parts or states of elements.

Practice1.html | practice2.html

practice2.html > html > head > style > ul

```
<!DOCTYPE html>
<html>
<head>
  <title>Pseudo-classes & Pseudo-elements</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f9f9f9;
      color: #333;
      padding: 40px;
    }

    h1 {
      text-align: center;
      color: #2c3e50;
    }
    h2::before {
      content: "🔹 ";
      color: #2980b9;
    }

    p::first-letter {
      font-size: 22px;
      font-weight: bold;
      color: darkred;
    }
    p::first-line {
      text-transform: uppercase;
    }
```

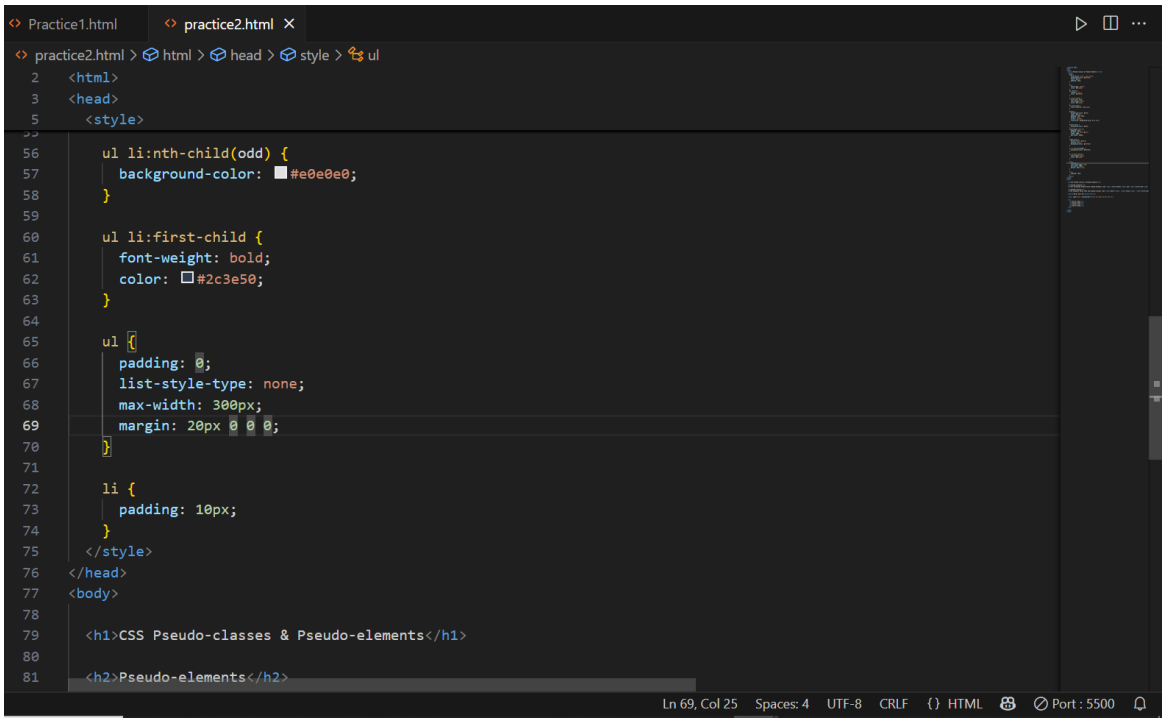
Ln 69, Col 25 Spaces: 4 UTF-8 CRLF {} HTML Port : 5500

Practice1.html | practice2.html

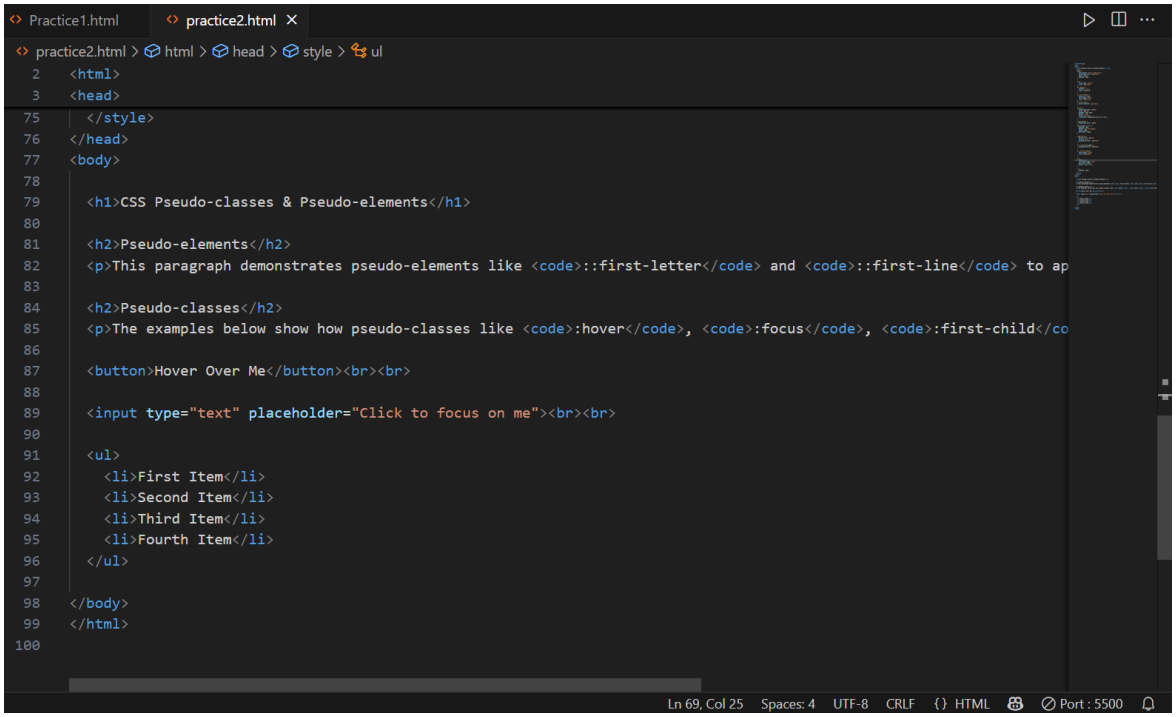
practice2.html > html > head > style > ul

```
<html>
<head>
  <style>

    button {
      background-color: #444;
      color: white;
      padding: 10px 20px;
      border: none;
      cursor: pointer;
      transition: background-color 0.3s ease;
    }

    button:hover {
      background-color: #666;
    }
    input[type="text"] {
      padding: 10px;
      border: 2px solid #ccc;
      width: 100%;
      max-width: 300px;
    }

    input:focus {
      border-color: green;
      outline: none;
      background-color: #e8f5e9;
    }

    ul li:nth-child(odd) {
```

Ln 69, Col 25 Spaces: 4 UTF-8 CRLF {} HTML Port : 5500



```
Practice1.html  practice2.html
practice2.html > html > head > style > ul
  <html>
  <head>
    <style>

      ul li:nth-child(odd) {
        background-color: #e0e0e0;
      }

      ul li:first-child {
        font-weight: bold;
        color: #2c3e50;
      }

      ul {
        padding: 0;
        list-style-type: none;
        max-width: 300px;
        margin: 20px 0 0 0;
      }

      li {
        padding: 10px;
      }
    </style>
  </head>
  <body>

    <h1>CSS Pseudo-classes & Pseudo-elements</h1>

    <h2>Pseudo-elements</h2>

Ln 69, Col 25  Spaces: 4  UTF-8  CRLF  {} HTML  Port : 5500
```



```
Practice1.html  practice2.html
practice2.html > html > head > style > ul
  <html>
  <head>
    </style>
  </head>
  <body>

    <h1>CSS Pseudo-classes & Pseudo-elements</h1>

    <h2>Pseudo-elements</h2>
    <p>This paragraph demonstrates pseudo-elements like <code>::first-letter</code> and <code>::first-line</code> to ap

    <h2>Pseudo-classes</h2>
    <p>The examples below show how pseudo-classes like <code>:hover</code>, <code>:focus</code>, <code>:first-child</code>

    <button>Hover Over Me</button><br><br>

    <input type="text" placeholder="Click to focus on me"><br><br>

    <ul>
      <li>First Item</li>
      <li>Second Item</li>
      <li>Third Item</li>
      <li>Fourth Item</li>
    </ul>

  </body>
  </html>

Ln 69, Col 25  Spaces: 4  UTF-8  CRLF  {} HTML  Port : 5500
```

Conclusion:

Through this task, I explored how CSS transitions and animations enhance visual dynamics by making style changes smooth and lively. Pseudo-classes and pseudo-elements further refine user interaction and control over content structure.