7-Day CSS Learning Plan (Detailed and Practical)

This plan ensures you cover both theory and practice daily for an hour.

Day 1: CSS Fundamentals

Topics to Cover:

- What is CSS? How it works with HTML.
- CSS Syntax: Rules, properties, and values.
- **Selectors**:
  - Basic Selectors: Element (`div`), Class (`.class`), ID (`#id`).
  - Grouping and Universal Selectors (`*`).
- **Basic Properties**:
  - Text: `color` , `font-family` , `font-size` , `text-align` , `line-height` .
  - Background: `background-color` , `background-image` .
- Adding CSS to HTML:
  - Inline, Internal (`<style>`), External (`<link>`).

Practical Task:

1. Create a simple webpage with a title, paragraph, and an image.
2. Style the text, set the background color, and add a unique font.

Tools & Resources:

- Google Fonts (fonts.google.com)
- MDN: CSS Basics

Day 2: Box Model

Topics to Cover:

- Box Model Components:
  - `margin` , `border` , `padding` , `content` .
- Width & Height Properties:
  - `width` , `height` , `max-width` , `max-height` .

- `border` : Width, style, and color.
- Overflow control: `overflow` , `overflow-x` , `overflow-y` .

Practical Task:

1. Build a card layout with a title, image, and description.
2. Use margins and padding to space elements properly.
3. Add borders with rounded corners.

Resources:

- MDN: Box Model

Day 3: Positioning and Flexbox Basics

Topics to Cover:

- Positioning:
  - `static` (default), `relative` , `absolute` , `fixed` , `sticky` .
  - Use `top` , `right` , `bottom` , `left` with positioning.
- **Flexbox Basics**:
  - `display: flex` and `flex-direction` (row, column).
  - Align Items:
    - `justify-content` (center, space-between, etc.).
    - `align-items` (center, stretch).

Practical Task:

1. Create a navigation bar with Flexbox.
2. Create a simple header with text centered vertically and horizontally.

Resources:

- CSS-Tricks: Flexbox Guide

Day 4: Advanced Layout with Grid

Topics to Cover:

- CSS Grid:
  - Define Grid Container: `display: grid`, `grid-template-columns`, `grid-template-rows`.
  - Placing Items: `grid-column`, `grid-row`, `gap`.
- Nesting Grids.
- Grid vs. Flexbox comparison.

Practical Task:

1. Create a 3x3 image gallery using CSS Grid.
2. Add spacing (`gap`) between images.

Resources:

- MDN: CSS Grid

Day 5: Pseudo-classes, Pseudo-elements, and Transitions

Topics to Cover:

- **Pseudo-classes**:
  - `:hover`, `:focus`, `:nth-child`, `:first-child`, `:last-child`.
- **Pseudo-elements**:
  - `::before`, `::after`, `::placeholder`.
- **Transitions**:
  - `transition` property: Add smooth animations for `color`, `background`, etc.

Practical Task:

1. Create a button with hover and focus effects.
2. Use pseudo-elements (`::before` and `::after`) to add decorative elements to a section.
3. Apply transitions to create smooth hover effects.

Resources:

- MDN: CSS Pseudo-classes

Day 6: Responsive Design

Topics to Cover:

- Media Queries:
  - `@media` for different screen sizes.
  - Breakpoints (`min-width` , `max-width`).
- Units:
  - Absolute (`px` , `pt`).
  - Relative (`%` , `em` , `rem` , `vw` , `vh`).
- Flexbox and Grid in Responsive Layouts.
- Mobile-First Design.

Practical Task:

1. Make an existing layout responsive using media queries.
2. Use percentage-based widths for a flexible design.

Resources:

- Responsive Design Basics

Day 7: Advanced Techniques and Project

Topics to Cover:

- CSS Variables:
  - Define and use variables (`--main-color` , `var(--main-color)`).
- Advanced Animations:
  - `@keyframes` and `animation` property.
- Combining Flexbox, Grid, and Media Queries.

Practical Task:

1. Build a portfolio or landing page:
   - Use Grid for the main layout.
   - Use Flexbox for internal components.
   - Add responsive styles with media queries.
   - Include animations for buttons and hover effects.

Bonus:

- Experiment with advanced pseudo-classes like `:nth-child(even)` and animations.

Resources:

- CSS Animation Guide

Tips for Success

1. Spend 10 minutes reviewing yesterday's topics before starting a new day.
2. Save all your practice files and revisit them to see your progress.
3. Use online sandboxes like CodePen or JSFiddle for quick experimentation.

Let me know if you'd like project ideas or additional resources for any specific day!