# Lab: CSS Grid + Flexbox Integration

**Learning Objectives**

By the end of this lab, students will:

* Understand when to use **Grid vs Flexbox** in real-world layouts.
* Build a responsive web page using Grid for layout and Flexbox for internal alignment.
* Combine both techniques for navigation bars, content cards, and footers.

## Step 1 – Starter HTML

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Grid + Flexbox Lab</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<div class="page">

<header>

<h1>My Portfolio</h1>

<nav>

<ul class="nav-list">

<li><a href="#">Home</a></li>

<li><a href="#">Projects</a></li>

<li><a href="#">About</a></li>

<li><a href="#">Contact</a></li>

</ul>

</nav>

</header>

<main>

<section class="intro">

<h2>Welcome</h2>

<p>This page combines CSS Grid and Flexbox for a clean, responsive design.</p>

</section>

<section class="projects">

<div class="card">Project A</div>

<div class="card">Project B</div>

<div class="card">Project C</div>

<div class="card">Project D</div>

</section>

</main>

<footer>

<p>

© 2025 My Portfolio | Follow me on

<a href="#">LinkedIn</a> & <a href="#">GitHub</a>

</p>

</footer>

</div>

</body>

</html>

## Step 2 – Base Grid Layout (style.css)

/\* Reset + general styling \*/

body {

margin: 0;

font-family: Arial, sans-serif;

background: #f8f9fa;

color: #333;

}

.page {

display: grid;

grid-template-areas: "header" "main" "footer";

grid-template-rows: auto 1fr auto;

min-height: 100vh;

}

/\* Grid areas \*/

header {

grid-area: header;

background: #34495e;

color: white;

padding: 15px 25px;

}

main {

grid-area: main;

padding: 20px;

}

footer {

grid-area: footer;

background: #2c3e50;

color: white;

text-align: center;

padding: 10px;

}

**Explanation:**

* The page is structured using Grid with three named areas: header, main, footer.
* Each section can contain internal layouts (where Flexbox will come in next).

**Step 3 – Flexbox Navigation Bar**

/\* Flexbox in header \*/

header {

display: flex;

justify-content: space-between; /\* title on left, nav on right \*/

align-items: center;

flex-wrap: wrap; /\* wrap if too small \*/

}

.nav-list {

list-style: none;

display: flex;

gap: 20px;

padding: 0;

margin: 0;

}

.nav-list a {

color: white;

text-decoration: none;

font-weight: bold;

}

.nav-list a:hover {

text-decoration: underline;

}

**Explanation:**

* Header uses Flexbox for **horizontal alignment** of title and nav.
* .nav-list uses Flexbox for **spacing** between menu items.
* flex-wrap makes the nav stack neatly on smaller screens.

## Step 4 – Grid for Project Cards

.projects {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(220px, 1fr));

gap: 15px;

margin-top: 20px;

}

.card {

background: #3498db;

color: white;

padding: 40px 20px;

border-radius: 8px;

text-align: center;

font-size: 1.1em;

display: flex;

justify-content: center; /\* centers text \*/

align-items: center; /\* vertically centers text \*/

transition: transform 0.2s ease;

}

.card:hover {

transform: scale(1.05);

}

**Explanation:**

* The project section uses Grid for a **responsive layout**.
* Each .card uses Flexbox to **center content** inside it.
* This combination gives clean structure (Grid) + precise alignment (Flex).

## Step 5 – Responsive Adjustments

@media (max-width: 700px) {

header {

flex-direction: column;

align-items: flex-start;

}

.nav-list {

flex-wrap: wrap;

gap: 10px;

}

.card {

font-size: 1em;

}

}

**Explanation:**

* At narrow widths, header stacks vertically.
* Navigation links wrap naturally due to Flexbox flexibility.
* The project grid automatically reflows because of auto-fit.

**Your Tasks**

1. Add a **sidebar** beside main using Grid (grid-template-areas).
2. Use **Flexbox** inside that sidebar to vertically align icons and labels.
3. Add a **hero banner** to the top of the page using display: flex to center text over an image.
4. Make the footer use Flexbox to space items evenly.

## Optional[[1]](#footnote-1) Extension

Add a “contact” section:

.contact {

display: flex;

flex-direction: column;

gap: 10px;

}

Then add inputs and buttons to practice alignment and spacing in a form layout.

**END**

1. [↑](#footnote-ref-1)