Final Portfolio Project: Step-by-Step Code Assignment

Perfect for beginners! Follow every step below to build a responsive Portfolio Page that links out to eight mini-projects. You'll learn semantic HTML, CSS Variables, Flexbox, CSS Grid, Media Queries, and GitHub Pages deployment. Let's make coding fun!

Learning Objectives

Semantic HTML5: Use tags like <header>, <nav>, <main>, <section>, <footer>.

CSS Reset & Variables: Always start with a reset; define reusable variables.

Layout Mastery: Build with Flexbox and Grid.

Mobile-First Design: Write base CSS for phones, then enhance for tablets/desktops.

Commenting & Organization: Keep code clean and documented.

Folder Structure & Linking: Organize files and link them correctly.

Deployment: Push your work live on GitHub Pages.

Fun Fact:
The first CSS Grid specification was published in 2011—but only became widely supported in 2017!

Prerequisites & Rules

Tools: A code editor (VS Code, Sublime Text), Git installed, a GitHub account.

Naming Convention:

Folders & files: lowercase + hyphens (e.g., product-landing/)

Classes/IDs: lowercase + hyphens (e.g., .site-header, #projects)

Indentation: 2 spaces per level.

Comments: Explain why, not what.

HTML: <!-- your comment -->

CSS: /* your comment */

Syntax Tip: Always save your file before switching to the browser to see live changes!

1. **Create Your Folder Structure**

Make a root folder named portfolio-final.

Inside it, create these exact items:

```
portfolio-final/
├── index.html
├── style.css
├── assets/            ← put images/fonts/icons here
├── tribute-page/        ← FCC: Tribute Page
│   └── index.html
├── survey-form/         ← FCC: Survey Form
│   └── index.html
├── product-landing/      ← FCC: Product Landing Page
│   └── index.html
├── tech-docs/           ← FCC: Technical Documentation Page
│   └── index.html
├── gooey-effect/         ← Custom demo
│   └── index.html
├── project1/           ← Your extra project
│   └── index.html
└── superhero-page/       ← Your superhero-themed page
    └── index.html
```

Placeholder: Each index.html in sub-folders can be empty for now.

Checkpoint: Double-check your tree! Missing or misnamed folders break your links.

2. **Base HTML Skeleton (index.html)**

Open portfolio-final/index.html and paste:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <!-- 1) RESPONSIVE META TAG -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <!-- 2) PAGE TITLE -->
  <title><!-- TODO: Your Name • Portfolio --></title>
```

```html
  <!-- 3) LINK GLOBAL STYLES -->
  <link rel="stylesheet" href="style.css" />
</head>
<body>

  <!-- STEP 4: HEADER & NAV -->

  <main>
    <!-- STEP 5: ABOUT ME -->

    <!-- STEP 6: PROJECTS GRID -->
  </main>

  <!-- STEP 7: OPTIONAL FOOTER -->

</body>
</html>
```

Breakdown

<!DOCTYPE html> → Enables HTML5 mode.

<meta charset="UTF-8"> → Ensures proper text encoding.

<meta name="viewport"> → Required for mobile responsiveness.

<title> → Shows in the browser tab.

Definition:
Semantic HTML uses tags that describe their content (e.g., <nav> for navigation).

3. CSS Reset & Variables (style.css)

Type at top of style.css:

```css
/* ==== 3.1 RESET & BOX-SIZING ==== */
*, *::before, *::after {
  box-sizing: border-box;  /* include padding/border in width/height */
  margin: 0;               /* remove default browser spacing */
  padding: 0;
}

/* ==== 3.2 CSS VARIABLES ==== */
```

```
:root {
  --clr-primary: #2c3e50;   /* Navy blue */
  --clr-accent:  #1abc9c;   /* Mint teal */
  --clr-bg:      #f4f7f9;   /* Off-white */
  --ff-main:     'Arial', sans-serif;
  --spacing:     1.5rem;
}

/* ==== 3.3 GLOBAL STYLES ==== */
body {
  font-family: var(--ff-main);
  background: var(--clr-bg);
  color: var(--clr-primary);
  line-height: 1.6;
  scroll-behavior: smooth;  /* bonus smooth scroll */
}
a {
  color: var(--clr-accent);
  text-decoration: none;
}
/* TODO: Add a:hover and a:focus styles */
```

Fun Fact

CSS Variables (a.k.a. custom properties) let you change your color palette in one place—no more hunt-and-replace!

4. Header & Navigation

4.1 HTML

Under <body>, before <main>, add:

```
<header class="site-header">
  <h1><!-- TODO: Your Name --></h1>
  <nav class="site-nav">
    <ul>
      <li><a href="#about">About Me</a></li>
      <li><a href="#projects">Projects</a></li>
      <li><a href="#contact">Contact</a></li>
      <li>
        <a href="https://your-resume.pdf"
           target="_blank" rel="noopener">
          Resume
        </a>
      </li>
    </ul>
  </nav>
</header>
```

4.2 CSS

```
/* ==== 4. HEADER & NAV ==== */
.site-header {
  background: var(--clr-primary);
  color: white;
  padding: var(--spacing);
  display: flex;                  /* Flex container */
  justify-content: space-between; /* spread name & nav */
  align-items: center;            /* vertical centering */
}
.site-header h1 {
  font-size: 2rem;
}
.site-nav ul {
  list-style: none;               /* remove bullets */
  display: flex;
  gap: var(--spacing);
}
.site-nav a {
  border-bottom: 2px solid transparent;
  padding-bottom: 2px;
}
.site-nav a:hover,
.site-nav a:focus {
  border-color: var(--clr-accent);
}
```

Syntax Tip

Flexbox: Use justify-content to control horizontal spacing, align-items for vertical.

Accessibility: Include rel="noopener" whenever you open links in a new tab.

5. “About Me” Section
5.1 HTML

Inside <main>, add:

```html
<section id="about" class="about-section">
  <h2>About Me</h2>
  <p><!-- TODO: Write 2–3 sentences about yourself here. --></p>
</section>
```

5.2 CSS

```css
/* ==== 5. ABOUT ME ==== */
.about-section {
  max-width: 800px;
  margin: 2rem auto;           /* vertical + center */
  padding: 0 var(--spacing);
}
.about-section h2 {
  display: inline-block;       /* underline width = text width */
  border-bottom: 4px solid var(--clr-accent);
  margin-bottom: 1rem;
}
```

Why inline-block?
By default, <h2> is display: block (full width). inline-block shrinks it so the underline hugs the text!

6. Projects Grid

6.1 HTML Starter

Below About:

```html
<section id="projects" class="projects-section">
  <h2>My Projects</h2>
  <div class="grid-container">
    <!-- FCC: Tribute Page -->
    <a href="tribute-page/index.html" class="grid-item">
      <img src="assets/tribute-thumb.png" alt="Tribute Page Preview">
      <div class="item-info">
        <h3>Tribute Page</h3>
        <p>HTML / CSS</p>
      </div>
    </a>
    <!-- TODO: Add blocks for:
         survey-form, product-landing,
         tech-docs, gooey-effect,
         project1, superhero-page -->
  </div>
```

```
</section>
```

6.2 CSS Starter

```
/* ==== 6. PROJECTS GRID ==== */
.projects-section {
  padding: var(--spacing);
}
.projects-section h2 {
  display: inline-block;
  border-bottom: 4px solid var(--clr-primary);
  margin-bottom: 1rem;
}
.grid-container {
  display: grid;              /* enables CSS Grid */
  gap: 1rem;
  /* TODO: replace with responsive formula below */
  grid-template-columns: repeat(3, 1fr);
}
.grid-item {
  background: white;
  border: 2px solid var(--clr-accent);
  border-radius: 6px;
  overflow: hidden;
  text-decoration: none;      /* remove underline */
  display: flex;
  flex-direction: column;     /* stack image + info */
}
.grid-item img {
  width: 100%;
  display: block;
}
.item-info {
  padding: 0.75rem;
}
```

Student Task

Change

`repeat(3, 1fr)`

to

repeat(auto-fit, minmax(250px, 1fr))

In a CSS comment, explain why auto-fit + minmax makes your grid shrink and grow on different screens.

Definition:
CSS Grid is a two-dimensional layout system—great for rows and columns!

7. Mobile-First Media Queries

At the end of style.css, add:

```
/* ==== 7. MEDIA QUERIES ==== */
/* Base styles = mobile first */
@media (min-width: 600px) {
  /* TODO: increase padding on .about-section & .projects-section */
}
@media (min-width: 900px) {
  /* TODO: bump up .site-header h1 font-size & .site-nav gap */
}
```

Fun Fact:
“Mobile-first” became a best practice around 2015 when smartphones overtook desktop usage!

8. Build Each Mini-Project Page

For each folder:

Copy the root index.html → folder/index.html.

Update in that copy:

<title> to your project’s name

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

Add at top or bottom:

```
<!-- Back to portfolio -->
<a href="../index.html">← Back to Portfolio</a>
```

Insert your project’s HTML/CSS/JS into <body>.

Comment major sections:

```
<!--
  Project: Survey Form
  Description: A responsive form to collect user data.
  Techniques: HTML form validation, Flexbox layout.
-->
```

Tip: Always test your relative paths by clicking each link in the browser!

9. Deploy to GitHub Pages

```
cd path/to/portfolio-final
git init
git add .
git commit -m "Final Portfolio"
git remote add origin https://github.com/YourUser/portfolio-final.git
git push -u origin main
```

On GitHub → Settings → Pages

Under Source, select main branch, root

Click Save → your site is live at

https://YourUser.github.io/portfolio-final/

Quick Git Glossary:

- git init: start a new repo
- git add .: stage changes
- git commit -m "msg": record changes
- git push: upload to GitHub

Submission Checklist

- ☐ ***Folder structure matches exactly (8 sub-folders + assets).***
- ☐ ***index.html & style.css are responsive (test at 320px, 768px, 1200px).***
- ☐ ***Eight .grid-item cards link correctly.***
- ☐ ***Each mini-project folder has a working, commented index.html.***

- [ ] ***Used CSS variables, Flexbox, Grid, media queries.***
- [ ] ***Code is indented, semantic, and documented.***
- [ ] ***Live on GitHub Pages; URL submitted.***