Project 16: Personal Portfolio Website

Dorian Knight

June 2025

Contents

1 Project Overview

The personal portfolio website is a bespoke collection of webpages coded using HTML, CSS and JavaScript, hosted using GitHub Pages. The website is a succinct distillation of my education, acquired skills and noteworthy projects. [Link to site] [Link to GitHub repository]

2 Purpose

The purpose of the personal portfolio website was to serve as a first step in building a personal brand. When applying to jobs there is often a place to put a link either to GitHub, Linkedin or a personal website. I've decided that I want to make a statement and showcase that I'm a hardworking talented individual with the tenacity to have a vision and execute a project through to the finish.
This website serves the following goals:

- Start building a reputation even before I've said a word or sent an email.
- Serve as a one stop shop for my personal links (GitHub, Linkedin, contact email, publications and resume).
- To allow visitors to efficiently filter through my portfolio of work to find a project proving I have what it takes to join their team.
- Show that I have initiative.
  - This is a hard skill to showcase on a resume but showcasing the results of my hard work will help to back up my resume and increase my credibility as a young engineer.
- Make it easier to send cold emails to research teams/engineering firms.
  - Instead of spending extensive time guessing how to best impress a company, I could get my message across quickly by indicating interest and linking to a project that demonstrates excellence on my website. Furthermore, the project that I directly link to will already have adequate documentation assembled to accompany the project description.
  - Having all the information in one place will also allow me to skip the guessing game of figuring out "what I should show them" and instead provide them with a starting point for which a visitor can branch out and learn more about who I am.

By encapsulating all of my work across various domains of engineering, I hope to better make the case why I'd be an excellent hire and a great addition to any team I join.

3 Outcomes

Using Figma, HTML, CSS, JavaScript and GitHub I have been able to design a functional website that allows visitors to learn the following:

- Who I am (about me page)
- My education (about me page)
- My driving academic interests (about me page)
- My skills (about me page)
- My work (portfolio page)

The portfolio is filterable. By using the "type" and "skills" filters the visitor can quickly find projects that prove that I have the skills they actively care about. While the website is always in flux, it will serve as a tool that can grow as I progress throughout my career showcasing my accomplishments and skill growth.

4 Skills Used

(a) HTML5 Icon

(b) CSS3 Icon

(c) JavaScript Icon

(d) GitHub Icon

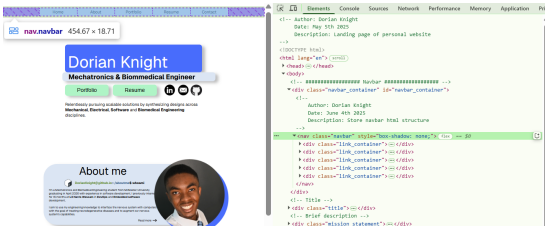
Figure 1: Skills used in project

HTML was used to provide a skeleton structure and label divisions within the viewable page. CSS was used to style, size and format these divisions to achieve an appealing aesthetic. JavaScript was used to make the website functional. By leveraging Asynchronous JavaScript And XML (AJAX) I was able to launch XMLHttp requests through the visitor's browser to fetch project information avoiding the need for a custom backend server. Launching AJAX requests allowed me to use my GitHub repository as the de facto server (more information in "Supporting Dynamic Interactions" section). GitHub was used as a centralized location to store all website code, project metadata, and project documentation.

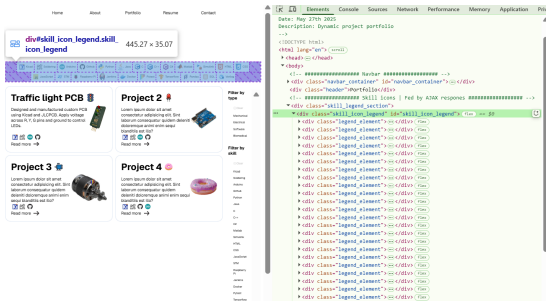
5 Static Structure

5.1 Use of display: flex

Setting a div's display type to flex was used throughout the website to achieve an information dense layout by allowing individual content elements to cascade across the screen horizontally instead of being forced to stack up vertically. The navbar and skills legend are examples where flex was used to increase information density.



(a) Navbar flex layout



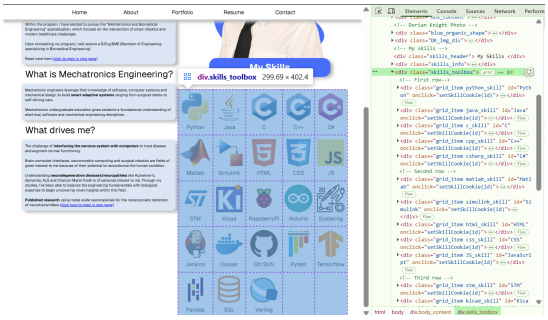
(b) Demo site skill legend flex layout

Figure 2: How flex layout was used in development

5.2 About me skill grid

The skill grid showcased on the about.html page holds the icons representing all of the skills featured in my project portfolio. The goal of the grid is to allow visitors (eg. prospective employers) to quickly find the skill they're looking for and see projects that showcase my use of said skill. By clicking on the skill icon the visitor will be automatically redirected to the portfolio page with a "skill filter" applied for the icon they clicked on.

The grid itself was coded using the "display: grid" style and utilizes "grid-template-areas" for the icon placement. Each grid item has a class embedded into its HTML code which is then referenced in the CSS style sheet where the icon is given a "grid-area". This grid area is then used in the parent div's CSS styling to order the skills.



(a) Demo site about me skill grid

```
.skills_toolbox {
    display: grid;
    grid-template-areas: "python    java      c         cpp       csharp      "
                         "matlab    simulink  html      css       js          "
                         "stm       kicad     rpi       arduino   soldering   "
                         "jenkins   docker    git       pytest    tensorflow  "
                         "pandas    sql       verilog   .         .           ";

    width: fit-content;
    position: absolute;
    right: 4vw;
    top: 62vw;
}
```

(b) Demo site grid template areas

Figure 3: Grid template areas ordering

As seen in 3b, the icon order in 3a's grid matches the CSS grid-template-areas attribute. This was done by design so similar skills could be easily grouped together (eg. Matlab next to Simulink / HTML, CSS and JavaScript grouped together on one line). This design also anticipates my skill growth over time. Say I learn PHP and Apache and I want all my web development skills to be displayed on a single row. With this grid template areas structure, I could easily move the HTML, CSS and JavaScript icons to the same line as PHP and Apache to show that I'm experienced in full stack web development without even having to say it. This increases information density and improves the user experience.

5.3 Portfolio project structure

Projects in the portfolio page are displayed in rows where each row contains two projects. Each individual project square is broken down into four sections: project title, project description, project skill images and the read more section. This structure allows me to lay out project information in a consistent way ensuring visitors can compare and contrast projects easily. This layout structure also makes it easy to dynamically populate project information using JavaScript when the visitor applies type and skill filters.

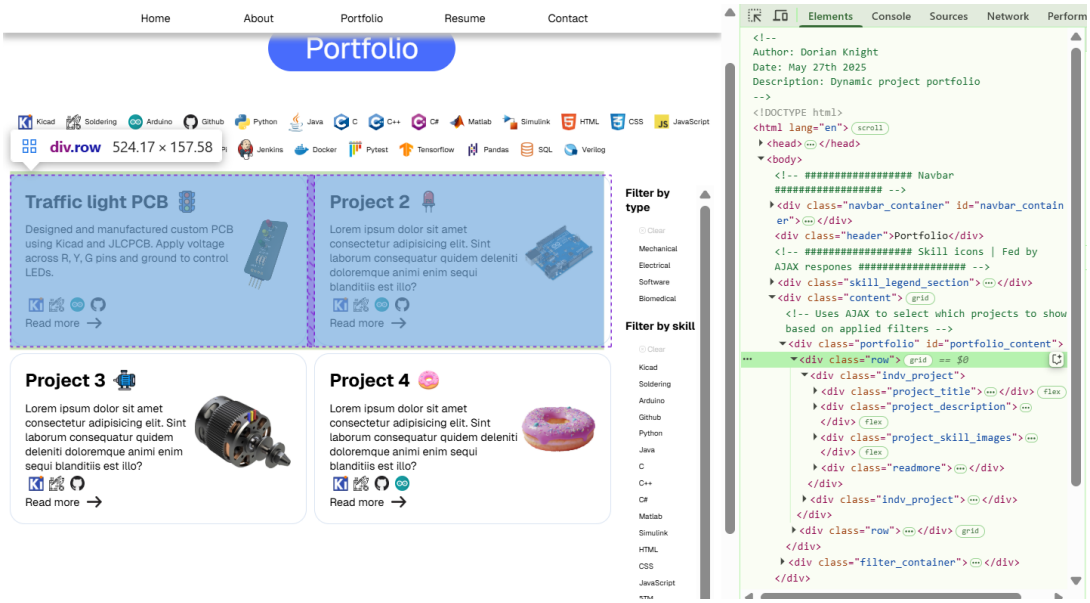
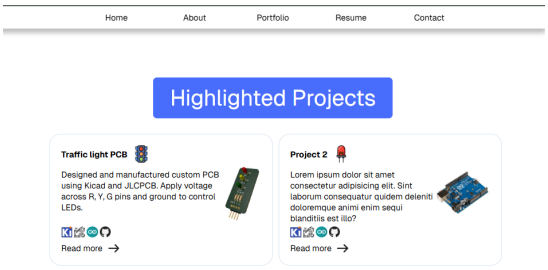


Figure 4: Portfolio row structure

5.4 Use of display: sticky

To improve the user experience a couple of useful divs are made to be sticky such that the visitor always has the ability to effectively navigate my portfolio website.

One of these divs that are sticky is the navigation bar:



(a) Sticky navigation bar

```
/* ############ Navigation bar styling ############ */
.navbar_container {
    top: 0vw;
    position: sticky;
    z-index: 1;
}
```

(b) Navigation bar container css

Figure 5: Navigation bar browser render and styling

Figure 5a shows the navbar at the top of the screen despite the visitor being more than halfway down the landing page. Having the navigation bar stick to the top of the screen as the visitor scrolls down the page allows for easy and continuous access to the other parts of the website improving overall user experience.

Sticky display is also used to keep the filter div near the top of the screen as the visitor is scrolling through projects. Within the filter div, sticky display is used again to keep the filter headers from scrolling out of the div ("filter by type" and "filter by skill" stay at the top of the div in figure 7). Keeping the headers visible gives the visitor context as they select their filters improving user experience.

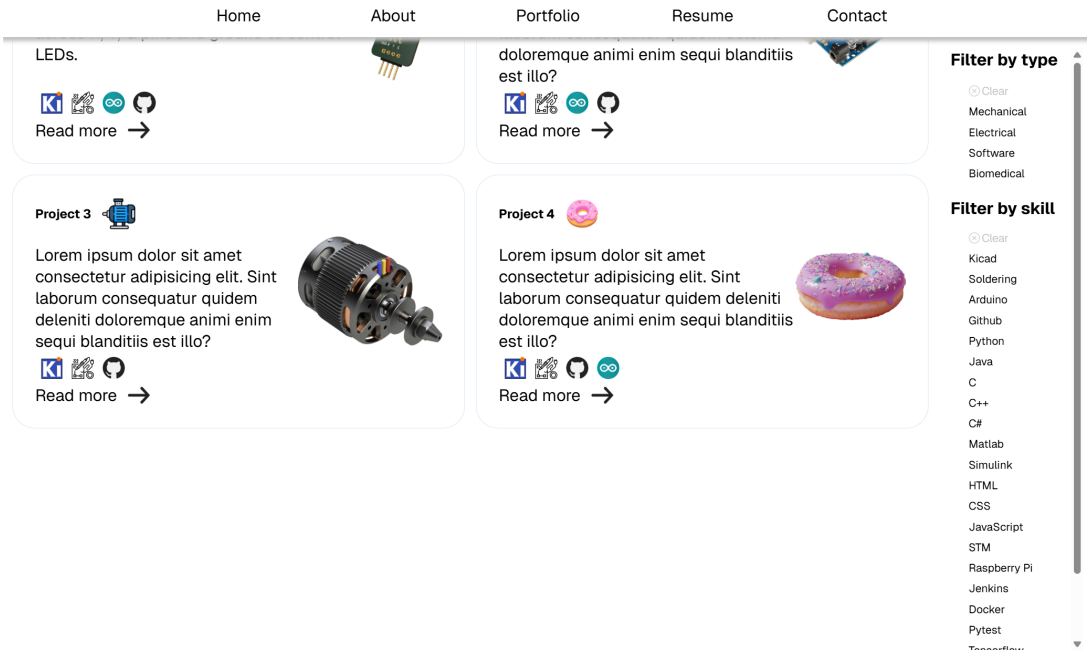


Figure 6: Entire filter div held just below nav bar to keep filters visible

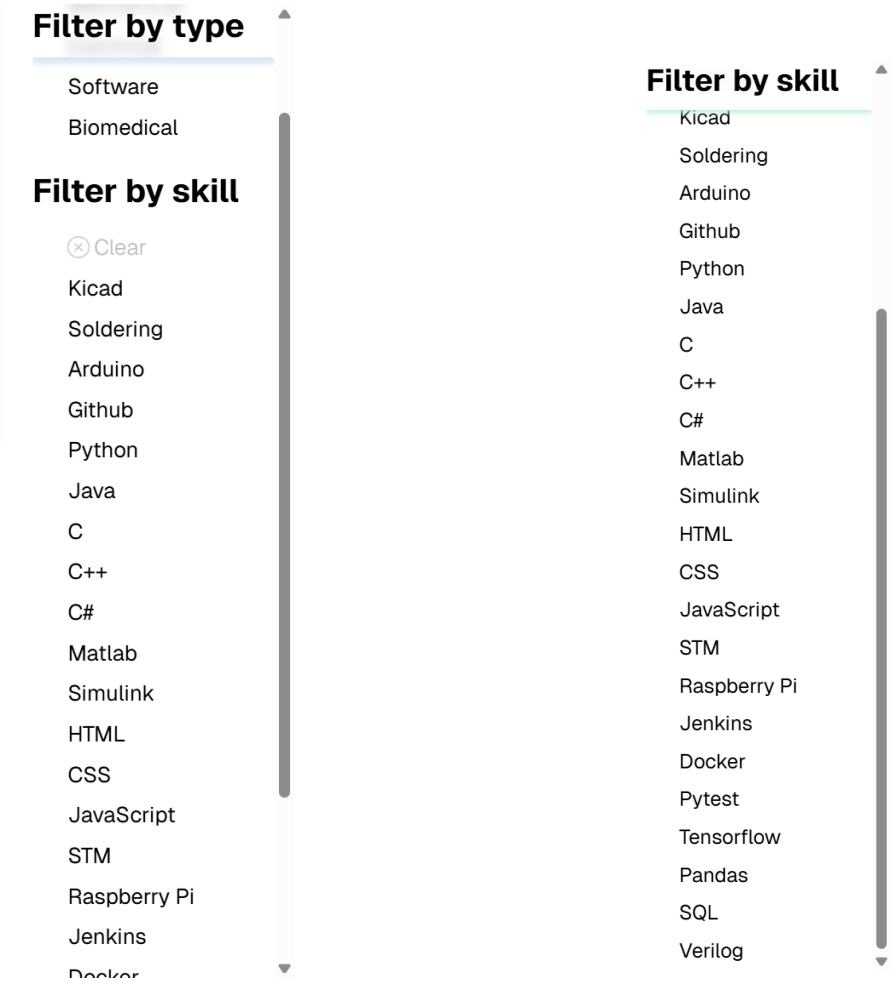


Figure 7: Headers within filter div kept from scrolling off screen

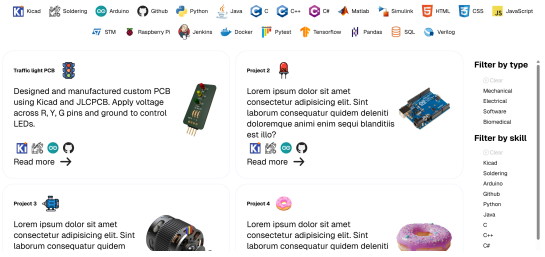
6 Supporting Dynamic Interactions

6.1 Filtering projects on portfolio page

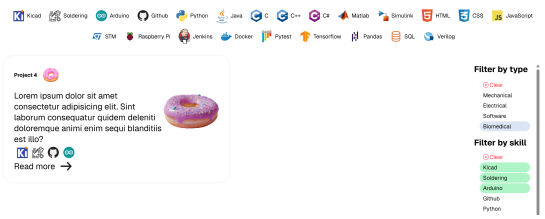
Being able to "filter" for projects with specific skills was a large functionality requirement when designing the portfolio page, and at first I had no idea how to satisfy the requirement. I started off with developing a backend server using LAMP (Linux, Apache, MySQL and PHP) with the expectation that I could populate the portfolio content with data stored in my SQL database. I eventually pivoted away from developing a backend server when I learned about Asynchronous JavaScript And XML (AJAX) requests. Using AJAX I launched my "queries" from the user's browser to my GitHub pages repository which hosted the .json files containing the portfolio data removing the requirement of housing the data in a custom LAMP backend server.
This had the effect of:

- Isolating the website's code from its content.
  - By holding the content in separate .json files I was able to avoid directly coding the project portfolio information into the JavaScript file itself which was my backup plan if the backend server development path failed.
- Avoiding the necessity of hard coding the filter tags and project descriptions in the portfolio.html file (in the case of a frontend only design)
  - HTML is programmatically generated in a JavaScript file based on the AJAX response and injected into placeholder divs contained within the portfolio page HTML file.

The end result was a set of modularized JavaScript files that fetched the portfolio jsons from GitHub, filtered the projects according to the selected tags applied by the visitor, and dynamically generated HTML showing the visitor all the projects that matched the tags they selected. All of which is accomplished without requiring a site reload.



(a) Demo portfolio with no filters applied



(b) Demo portfolio filtered for Biomedical projects demonstrating skill with Kicad, Soldering and Arduino

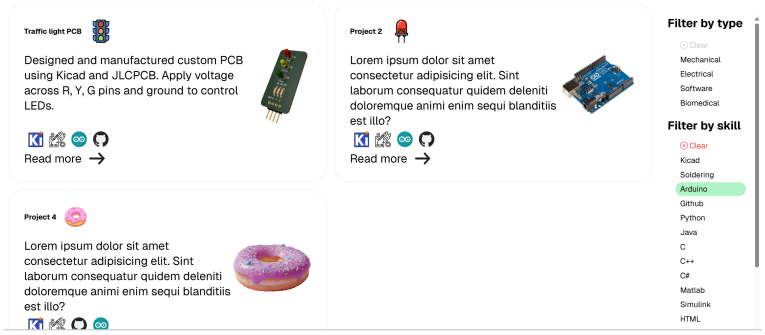
Projects can be filtered by "type" and by "skill". Project types are meant to provide visitors (eg potential employers) with general categories to quickly see the kind of projects that would most likely align with the skills required to perform well in their industry.

The skill filter is meant to be used as a fine filter and help visitors narrow the project list down even further. While all my software projects will be within the "Software" type filter, some software employers will care deeply about my Python experience and others would rather see my Verilog abilities. Having each skill advertised in the about me page made a filter in the portfolio page allows visitors to find what they want faster and improves the user experience.

6.2 About me — Portfolio filter interaction

Another critical requirement when designing the portfolio page was that it should dovetail nicely with the skill grid of the "About me" page. The visitor should be able to see a skill that interests them and then immediately be presented with all work that involved that skill. By using cookies in JavaScript I'm able to save a cookie with information describing the skill clicked on from about.html. Then when redirected to portfolio.html, I apply a filter corresponding to the saved cookie skill value. This seamless filtering allows visitors to quickly get the information they want and improves the user experience.

(a) Select Arduino from about me skill grid



(b) Arduino filter applied in "filter by skill"

6.3 Modularizing common HTML structures

While working on the website I realized that the navigation bar HTML structure would be common across all pages and thus should likely be imported instead of repeated on each HTML page. While HTML can link to multiple style sheets and scripts allowing for CSS and JS modularization, there isn't a "quick" way to directly tie in other file's HTML into a parent HTML file. To get around this issue I created a navigation bar HTML file that would store the navbar structure, uploaded that file to my GitHub repository and queried for that file in navbar.js using AJAX which would write the HTML string into an already created "navbar placeholder" in the parent HTML file. By linking the same JavaScript code onto each HTML page, I was able to commonize the navbar HTML file such that if changes were to ever be made on the navbar, they only need to be made on one file.

7 Conclusion

I successfully made a fully functioning frontend website to showcase the work I've done and will do in the future. This website will serve as a supplement to my resume while searching for jobs and building my career. I'm glad I took the time to understand the basics of HTML, CSS and JavaScript instead of directly jumping into a framework (Eg React/Angular) without understanding what was going on behind the scenes. The next step is to fill the website with amazing awe-inspiring projects that will set me apart and prove that I'm a quality hire.