Researching modern web technologies

How can I make sure my portfolio is technologically solid?

Date	**:**	**18-09-2023**
Version	**:**	**1.1**
Status	**:**	**Definitive**
Author	**:**	**Luc Swinkels**

Version

Version	Date	Author(s)	Amendments	Status
0.1	12-09-2023	Luc Swinkels	First draft	Draft
1.0	13-09-2023	Luc Swinkels	First definitive version	Definitive
1.1	18-09-2023	Luc Swinkels	Updated document contents	Definitive

Table of contents

1. Context

The goal of this document is to help me build my portfolio by answering one of my sub questions:

“How can I make sure my portfolio is technologically solid?”

This means I need to figure out which technologies I am going to use to build my portfolio, as well as how I am going to perform performance tests.

2. Method

To answer this research question, I am going to use multiple library research CMD methods.

2.1 Literature study

I will be browsing the internet for resources on popular technologies, what their pros and cons are, and ultimately what would fit best with my type of project (building a portfolio).

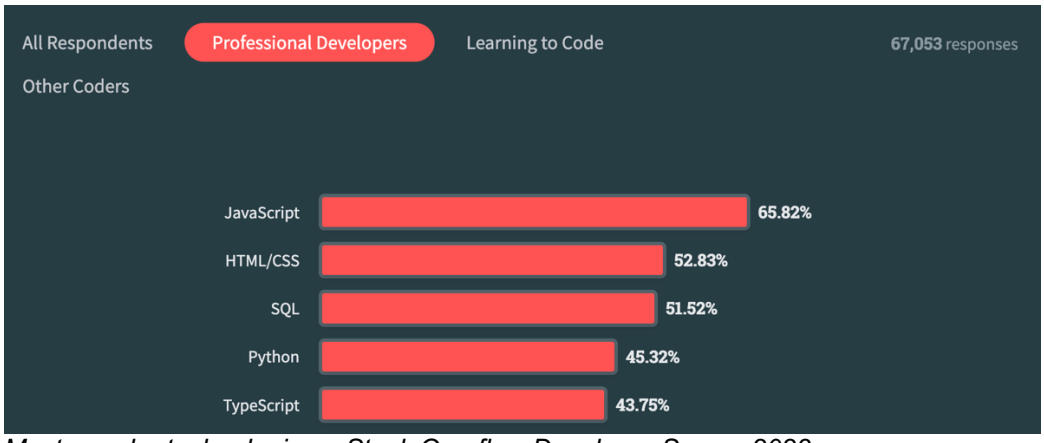
2.2 Trend analysis

I'm going to look at multiple trends in the web development space by using various community platforms in the webdevelopment space such as StackOverflow and Twitter and analyzing which technologies are often used by professional developers.

3. Results

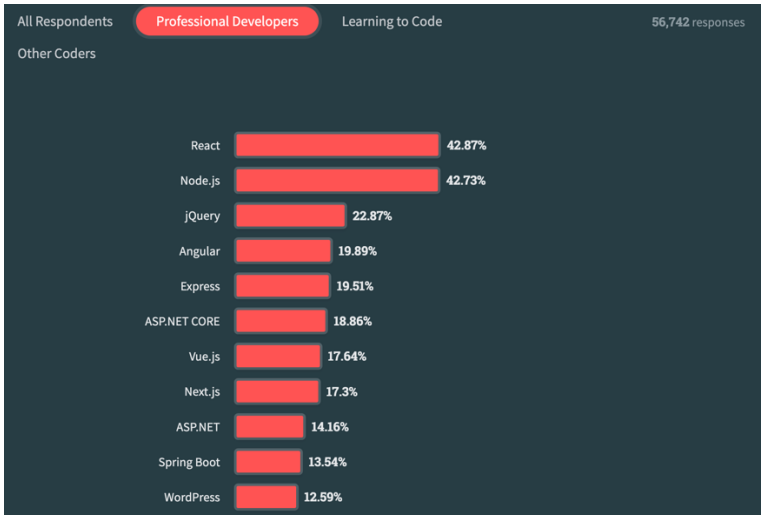
3.1 Stack Overflow 2023 survey

My first point of reference is going to be the 2023 Stack Overflow Developer survey (*Stack Overflow Developer Survey 2023*, 2023). This survey, answered by 90,000 developers, gives tons of insight in modern web development practices. I will be basing my findings on the professional developers survey results.



Most popular technologies – Stack Overflow Developer Survey 2023

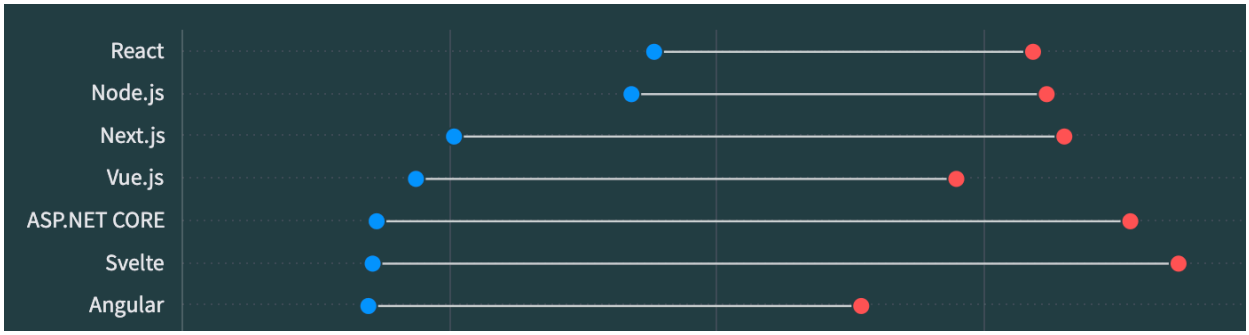
The obvious winner here is JavaScript, as TypeScript is a direct superset of it, and both of them are in the top 5 of most popular technologies (not just in the web field).



Most popular web frameworks – Stack Overflow Developer Survey 2023

For web frameworks, we also see a clear favorite in React, but we also see JavaScript has completely taken over the world of front-end web frameworks with React, Angular, Vue and Next.js.

Since I also care about development experience and not just what the “most popular” framework is, I also want to look at how much developers prefer working with a specific framework.



Most desired and admired technologies – Stack Overflow Developer Survey 2023 (Blue = desired, red = admired)

This graph shows that although frameworks like Svelte and Next.js aren’t as popular as React, they do prefer to work with them.

3.2 Framework

I’ve already decided I want to use a JavaScript based web framework because not only is it a very popular option according to the survey, I also already have experience with React and I don’t have the time to learn a different programming language in time (nor do I want to at this time, as React is still the industry standard).

3.2.1 React vs Next.js

The choice will be between React or Next.js as I have already worked with these therefore being biased in favor of them. Perhaps in the future I will look at other frameworks like Vue or Svelte, but for this project I want to use something I am somewhat familiar with.

But which one should I pick, and why? I’ve looked at the survey results as well as the features for each option on their official docs (React documentation) (Next.js documentation) and came up with the following pros and cons:

React

Pros	Cons
Industry standard	“Bad” SEO
The most popular JavaScript web framework	CSR only
	Somewhat outdated docs

Next

Pros	Cons
Better SEO	Redundant built-in features that I won’t need
Built-in API’s like image optimization & routing	
Built as a React framework, so any normal React code works	
Very organized docs	
CSR + SSR	

Since Next is built on top of React and it offers features like better SEO, built-in image optimization and simpler routing, it makes sense to me to use Next over React. While things like good SEO practices and

SSR aren't something I need to develop this portfolio, I might want to expand this portfolio in the future and having the option to use these built-in features is a big plus for me.

The docs for React are also often outdated because they are fast-developing, while Next.js docs are constantly updated as mentioned in an article by tech website Ninetailed (Ekainu, 2023). This helps me navigate through issues during the development process by being able to use reliable documentation.

3.3 Styling

For styling, I felt like a good starting point was to check the Next.js docs for options. They provide you with 5 options (*Building Your Application: Styling*, n.d.):

- Global CSS: Simple to use and familiar for those experienced with traditional CSS, but can lead to larger CSS bundles and difficulty managing styles as the application grows.
- CSS Modules: Create locally scoped CSS classes to avoid naming conflicts and improve maintainability.
- Tailwind CSS: A utility-first CSS framework that allows for rapid custom designs by composing utility classes.
- Sass: A popular CSS preprocessor that extends CSS with features like variables, nested rules, and mixins.
- CSS-in-JS: Embed CSS directly in your JavaScript components, enabling dynamic and scoped styling.

Out of these options, I have some previous experience with all of them, therefore I am able to judge myself what the right pick is for this project. Recently I've been enjoying using TailwindCSS for component based apps (which this project will also be using), therefore I think TailwindCSS is the best option. It gives me predefined utility classes which speeds up my development process by a lot, while still providing me with the freedom of writing my own CSS (instead of using a pre-built UI).

3.4 Hosting

As this portfolio will be.a fairly "static" website/web app where I serve static content to the user, Im opting to not use a back-end service to store data in, so I have free reign of where I want to host this project.

An article by C. Lewis written for HostingAdvice shows that there are a ton of web hosting providers for NextJS apps (*Lewis, 2023*). However, since I am only hosting a front-end service, I am mainly interested in free providers.

The three most popular options, as listed in the article are Vercel, Heroku and Netlify. Heroku used to be a powerful hosting provider but they have recently removed their free plan, and since Vercel is the company behind Next.js, I trust them more than Netlify to host my Next.js app as they offer seamless integration and easy deployment for Next.js apps, and the official Next.js docs also recommend using Vercel as a deployment platform to do this, although obviously biased (*Learn | Next.js*, n.d.).

3.5 Performance testing

Google offers tools like Lighthouse and Pagespeed Insights to test and analyze website performance. These tools are popular and often used by SEO experts to analyze their websites, as seen in an article from S. Connelly for seoClarity (*Connelly, 2023*). These tools have features such as checking page loads, image optimization, client-side file size distribution, which is all I need for a project of this size (*Google Lighthouse – SEO Glossary | SearchMetrics*, 2020).

3.6 **Code quality**

Code quality is important in maintaining a well-developed app that is scalable. N. Raval has written a detailed article for Radixweb on why it could be benefitial to use TypeScript over JavaScript (*Raval, 2023*). Some of the key features TypeScript has over JavaScript are:

- Catch possible bugs in your code while developing, instead of at runtime
- Increased intellisense during the development process by using type definitions

4. Conclusion

4.1 Framework

For the front-end framework I have chosen Next.js. It's quickly becoming the industry standard and is constantly improving. I could've chosen React, but I prefer the built-in optimization Next.js gives me such as Image optimization/lady loading, static page generation for better SEO, and SSR.

4.2 Styling

To style my app I will be using TailwindCSS, because I enjoy using a component based development structure and TailwindCSS is the industry standard to do this, by providing me with predefined utility classes that I can freely use in any way, leaving me with freedom to style the app however I like, but speeding up my development process by a lot.

4.3 Hosting

For deployment, I have chosen to use Vercel because it has seamless integration for Next.js apps and will make my development experience easier by providing me with out of the box CI/CD for production and testing environments by simply linking my portfolio's GitHub repo to Vercel.

4.4 Performance testing

Performance testing can only be done once (a prototype of) the website is up and running. Throughout the development process I will perform tests to check if my portfolio is well optimized and fast by using tools like Google Lighthouse and PageSpeed Insights, as they offer free insightful features like loading speeds, content serves, and targeted recommendations.

4.5 Code quality

I will be using TypeScript over JavaScript to build my Next.js app, because I think the features that help me catch bugs during the development process are very valuable. I also believe TypeScript is becoming the industry standard and will soon be just as popular as JavaScript.

5. Literature

1. *Stack Overflow Developer Survey 2023*. (2023, May). Stack Overflow. Retrieved September 12, 2023, from https://survey.stackoverflow.co/2023/
2. Ekainu, G. A. (2023). Next.js vs React: The Difference and Which Framework to Choose. *ninetailed.io*. https://ninetailed.io/blog/next-js-vs-react/
3. *Building your application: styling*. (n.d.). Next.js. Retrieved September 18, 2023, from https://nextjs.org/docs/app/building-your-application/styling
4. Lewis, C. (2023). 7 Best Next.js hosting Providers (Sep. 2023). *HostingAdvice.com*. https://www.hostingadvice.com/how-to/best-nextjs-hosting/
5. Raval, N. (2023, July 17). TypeScript vs JavaScript: Which One Should You Choose? *Radixweb*. https://radixweb.com/blog/typescript-vs-javascript
6. Connelly, S. (2023, February 27). *Top Page Speed Testing Tools For Analyzing Site Performance*. seoClarity. https://www.seoclarity.net/blog/best-page-speed-testing-tools
7. *Google Lighthouse – SEO Glossary | SearchMetrics*. (2020, September 29). Searchmetrics. https://www.searchmetrics.com/glossary/google-lighthouse/
8. *Learn | Next.js*. (n.d.). https://nextjs.org/learn/basics/deploying-nextjs-app/deploy