



Building an **Effective** **Dev Portfolio**

Josh Comeau

Building an Effective Dev Portfolio

Written by Joshua Comeau

© 2020 Joshua Comeau.

All rights reserved. The author has taken care in preparation of this book, but makes no expressed or implied warranty of any kind, and assumes no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in with or arising out of the use of the information contained herein.

This book is self-published.

INTRODUCTION

According to Career Karma[†], about 34,000 people graduated from software-development bootcamp programs in 2019, in the US alone. This is in addition to all of the folks who self-taught through online programs like FreeCodeCamp, as well as folks graduating from college with Computer Science / Software Engineering degrees.

That's a lot of people competing for junior software development jobs!

In order to land an offer, you'll need to stand out from the crowd. A college degree from a prestigious university is one way to do this, but not everyone is privileged enough to be able to attend one. Prior work experience is another way, though there's a catch-22 here: if you need work experience to get a job, how can you possibly get work experience?!

Without a degree or previous work experience, **your portfolio** is the greatest asset you have. A couple of well-executed projects sends *such* a strong signal to potential employers. Portfolio projects show that you have the skills required to do the job.

We need to do some work to showcase those projects, though. Even the most amazing portfolio project might be overlooked if an employer only sees a couple screenshots and a Github link. We need to *guide* potential employers through our work, making sure to highlight the most interesting, impressive, salient parts. The good stuff. Stuff that isn't always obvious at first glance.

This book is an instruction manual for creating a portfolio site that stands out to prospective employers, in order to give you the best chance of landing an offer. It focuses on the practical tips and tricks I've learned over the years to help capture the attention of the folks who review applications.

[†] Source: <https://careerkarma.com/blog/bootcamp-market-report-2020/>

MOTIVATION

A few months ago, I offered to do volunteer portfolio reviews on Twitter, and got over 200 responses (<https://twitter.com/JoshWComeau/status/1227213590274486275>).

After reviewing a whole bunch of these portfolios, I realized I was giving the same feedback in 75%+ of cases. Most junior developer portfolios—even ones that were very well-built—were missing a lot of potential opportunities to stand out to employers.

I know about what employers are looking for because I've been involved in hiring for multiple organizations, including Khan Academy, Unsplash, Breather, and more. I've worked with HR departments to decide whether to interview someone or not. I've been on both sides of the table.

I've been using this information to help aspiring developers for years. I work with Concordia University (through a partnership with Journey Education) to help coach recent graduates and ensure that their applications are as polished and effective as possible.

For a while now, I've wanted to share this information more broadly. I had initially planned to write a blog post, but I quickly realized I had a lot to say about it! So instead, I've written a short book.

I'm passionate about this topic because I feel like it's so often underappreciated. While the portfolio site is just one piece of the job-hunting puzzle, it's a piece that can have an outsized impact. I don't want to oversell it—even the best portfolio site won't get you a job all on its own—but I've seen how a solid portfolio site can lead to more callbacks, more interviews, and ultimately more job offers. The goal of this book is to help you build a stellar portfolio that can increase your odds of success.

CHAPTER I: FOUNDATIONS

WHAT IS A DEVELOPER PORTFOLIO?

A developer portfolio is a website that showcases the work that you've done as a developer. The idea is borrowed from the art world: photographers, for example, will often create a portfolio website that showcases their best work.

The goal of a developer portfolio is to present yourself and your work in the best possible light. Your portfolio should make employers excited to meet you. It should offer a glimpse into how awesome it would be to work with you!

Unless you already have extensive work experience, your side projects are the greatest asset you have. They are your trophies, and the portfolio site is the trophy case that highlights all the amazing stuff you've done.

Building side projects is a lot of work, and we want to make sure to squeeze as much job-seeking value out of them!

WHY DO I NEED ONE?

You may be wondering if you actually *need* a developer portfolio. You have a resume, and a LinkedIn profile. You can list your projects there, isn't that enough? Will employers actually take the time to visit a portfolio site?

Without a doubt, resumes and LinkedIn profiles matter, but I believe that a developer portfolio can be a kind of *secret weapon*.

LinkedIn and resumes are both focused on *work* experience. If you're early in your career, you don't have very much relevant experience. And while LinkedIn does let you add details about individual projects, the format makes it very difficult to share all the context. As we'll see, presentation matters, and the presentation on LinkedIn is less than ideal.

Portfolio sites let us guide potential employers through our projects, making sure they realize how *freaking cool* the stuff we built is, and how *relevant* our skills are for the roles

they're trying to fill. They allow us to make a strong case for our competence and experience.

Now, not all employers will take the time to look at our portfolio site. And, believe it or not, **this is a blessing in disguise.**

Most developers either don't have a portfolio site, or they haven't put much thought or effort into one. The reason for this is that "build a solid portfolio site" is rarely a top priority for recent bootcamp grads, or self-taught folks. Many developers skip this step, since they see it as optional.

We can imagine an alternative reality, one in which every developer has a polished portfolio, and every employer scrutinizes them rigorously. In this reality, it would be very hard to stand out from the crowd. We'd have to put in a *bunch* of work just to meet the minimum expectation! It wouldn't give us an *advantage* at all.

Happily, we don't live in that reality. In ours, *many* employers will check a portfolio site, but not all. For those that do, we'll have a huge advantage.

Unfortunately, job-seeking is competitive by nature; for every junior dev job-posting, there will be multiple candidates, and only one of them will get an offer. A polished, effective portfolio site can be just the leg-up you need.

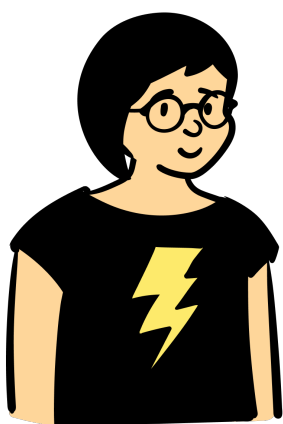
A portfolio site is a guided tour of your personal projects, a chance to show (rather than tell) that you have the skills needed for the job. Imagine if you're the only candidate to have a portfolio site listed with your application! I expect it'd be a serious competitive advantage.

Like I said: portfolio sites can be a secret weapon.

WHO IS THIS BOOK FOR?

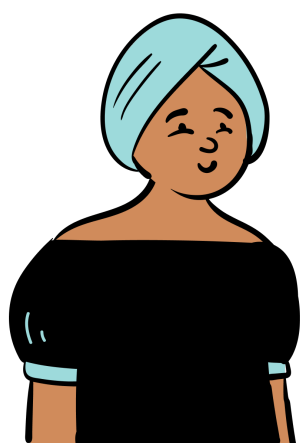
I wrote this book for people who are trying to break into the tech industry, or who are looking to make early moves in their career. It assumes that you've been writing code for long enough to have built a couple projects, but not long enough to have an impressive work history. It assumes that you're looking to get a job working as a front-end or full-stack software engineer.

As I was writing this book, I kept 4 distinct personas in mind:



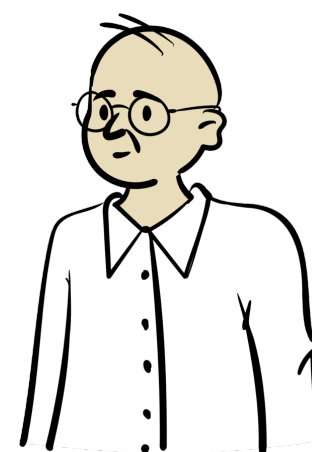
Morgan recently graduated from a 12-week coding bootcamp in Toronto. They have spent a few weeks polishing their final project from the course, and have started applying to positions.

Darius has been working as a junior front-end software developer for about 6 months in Austin. Unfortunately, he isn't enjoying his current role, and is looking to make a change. As he prepares to enter the job market again, he wants to give himself an edge by developing a solid portfolio site.



Priya is in her final year of Computer Science at UC Berkeley. She's participated in several hackathons and is particularly interested in full-stack development. She's contributed to several popular JS libraries. She's eager to get a job at a small start-up where she can have a large impact.

Levi is an office worker in Montreal who has been teaching himself web development through free online resources with React. It's been over a year of nights-and-weekend study, and he's even done a couple small freelance projects. To help with his office work, he created a small desktop application with Electron that lets him process invoices. He doesn't know if he's learned enough, but he's eager to change careers.



These people are all *early in their software development careers*. They all have a *body of work* they can share, and they're all hoping to *get a new job* working as a software developer.

I'm also focusing this book on front-end / full-stack software developers, but I expect that most of this content will also be relevant for back-end developers, mobile developers, or data engineers.

Caveats and Exceptions

- Software developers live and work all over the world, but I'm only familiar with hiring practices **in the United States and Canada**. If you're looking for a job in another part of the world, please understand that there may be differences that I am not aware of. Be skeptical of anything that doesn't sound right to you.
- This book assumes that you're **looking to get a job** as a software developer. If you're more interested in freelancing or starting your own company, the advice in this book is likely all wrong.
- If you don't yet have at least **1 solid personal project**, you should start with that. See the next section to get ideas for what to include.

CURATING PROJECTS FOR YOUR PORTFOLIO SITE

A portfolio site is a *showcase* of your work. But how do you know whether a project is worth showcasing? What sorts of projects qualify?

Ultimately, our goal is to show employers that we're competent at building stuff. There's a whole lot of different forms that this proof can take. Here are some examples, using our personas from earlier:

- **Morgan** spent 2 weeks at the end of their bootcamp building a restaurant reservation webapp, and they spent a few days afterwards tinkering and polishing it. The app is very much an MVP and it's missing a lot of features, but Morgan learned a lot building it.

Since graduating, Morgan has been volunteering at Code Dojo, a non-profit that runs bi-weekly workshops teaching programming fundamentals to kids. They've helped rework some of the curriculum.

- During the early stages of the Coronavirus pandemic, **Darius** spent a weekend building a Twitter bot using Ruby. It looks for people saying that coronavirus is “just like the flu”, and automatically replies with links to articles that clarify this misunderstanding.

Darius also spent 3 months in his first job contributing to a team that rolled out a notifications system.

- **Priya** has participated in several hackathons. For example, one time she teamed up with 3 other people to create an analytics platform that respects user privacy (A Google Analytics alternative with no GDPR banners!). She tackled the back-end side, ensuring that data is collected and processed effectively. The team completed a proof-of-concept, but have not taken any further steps to produce a real application.

Priya has also contributed to several JS libraries, like PancakeJS and em-dash.



- **Levi** created a small Electron app to help him scan and catalog invoices at his office job. It was never released to the public, since it's tailor-made for his specific use-case, but he uses it every day, and has saved him about 2 hours of work each week.

Levi also built a small single-page website for his local bakery, so that they could post their baking schedule, so that customers can always get fresh pies. 🥧

We're only scratching the surface — all kinds of different work qualifies. Morgan's volunteer work at a non-profit absolutely counts, even though there's no live demo or code samples. It still sends a strong signal about competence. Same thing applies for things like speaking at local meetups, being involved in the community, etc.

In my experience, certain kinds of projects are *especially* high-impact:

- ★ Did you build a project to solve a specific niche problem you had, like Levi's invoice-scanning app? Employers love to hear about projects like this, because it shows that you're able to apply your skills in creative ways, and tackle a project from start to finish.

-  Is your project “alive”? Is it shipped, and are people really using it for its intended purpose? It’s better to have a living, breathing project with real users (even if it’s only a handful) over a project that was built exclusively as a demo / for the portfolio.
-  Have you contributed to a popular open-source library? Participating in the open-source community builds many of the “soft skills” that employers treasure! In order to get your change merged, you’ll need to be an effective communicator, be able to estimate tasks, and be good at receiving feedback.

If you’ve contributed to open-source, definitely include it as a project! Just be sure to be very explicit about what your contributions were. Non-technical contributions, like adding documentation, are super valuable, and absolutely worth adding to your portfolio, but be careful; you don’t want to imply that you’ve contributed in ways that you haven’t.

If you don’t have any projects that fit these criteria, that’s OK! This list is meant to help people decide how to prioritize existing projects. It isn’t meant to exclude other kinds of projects.

PROJECTS NOT TO INCLUDE

Tutorial / Workshop exercises

Don’t include projects that were created by following a step-by-step tutorial. Same thing goes for workshops or exercises assigned by a bootcamp.

There are a few reasons for this:

- Projects are meant to show how you can use the skills you’ve accrued to build things in a self-directed, unguided way. Tutorials are a way to give you those skills, but the tutorial itself is not a demonstration of those skills.
- If you try to pass a tutorial project off as your own, this can backfire if the screener recognizes the project (more likely than you’d think!).
- Projects should be an accurate representation of where your skills are at. If you get hired on the belief that a tutorial project was your own invention, you’ll be assigned

work that could be well beyond your current skill level. This is a stressful nightmare scenario!

If you made *significant* extensions to the project, it might be alright to include. For example, if you were given an assignment to create a Tic-Tac-Toe game, and you decided to spend a week adding online multiplayer support. Or if you transformed it into a Sudoku game. A good rule of thumb: did you spend >50% of the *total time* on the extensions? If the bulk of your time was spent adding self-directed extensions, then I think it's worth adding. Just be sure to be explicit about which parts were original.

Non-Dev Projects

In an attempt to “pad” the number of projects, I’ve seen some folks add information about non-development projects like photography or carpentry.

There is a place for these kinds of details on your portfolio, but it shouldn’t live amongst your other projects. You can include personal information in your About Me section.

Confidential Work

If you have previously worked as a junior dev / completed an internship, you may be tempted to include features that you worked on in that capacity. This can absolutely be a great idea, but you need to be careful and ensure that you’re able to talk about it. Especially if you work for an agency, you can get in trouble; agencies often do work under a non-disclosure agreement.

When in doubt, ask your employer for clarification.

The Portfolio Site Itself

Some developers will list their portfolio site as a project on their portfolio site. In general, I believe that it’s better not to do this; it gives the impression that you’re filling space because you don’t have other projects you can use.

The exception is if you invested time into parts of the site that are non-obvious from a user perspective. For example, did you build an “admin panel” to manage the content, like a mini home-built CMS? Did you code an analytics page to learn about traffic? Absolutely include it in this case!

HOW MANY PROJECTS TO INCLUDE?

At least 2. No more than 5.

Something which can be surprising: It's better to have 1 large polished project than 5 small projects.

You want to show that you can complete a non-trivial project from start to finish. Depth is more important than breadth, because it shows that you have the grit and determination to stick through the hard parts and finish a large project.

What if I have more than 5?

If you have a dozen projects that you feel are worth including, it's likely that you've gone *too wide* and *not deep enough*. Pick the 5 projects that best represent your skills and the work you want to do, and (if possible) consider spending a few days extending your largest project to be even larger.

Nobody will spend time looking at more than 5 projects. Most people will only look at 1 or 2. If you include 12 projects, a potential employer might only look at your least impressive one!

Portfolios are meant to highlight your *best* work, they aren't meant to be an exhaustive set of *all* of your work.

If you really want to include more than 5 projects, tuck the other projects behind a "show all" or "view archive" link. That way, you're setting a clear distinction between the "top-shelf" stuff and everything else.

What if I only have 1 project?

If you only have 1 significant project, you have a few options:

- Spend a weekend on a smaller second project. Consider building something relevant to your interests; if you're really into yoga, for example, build something that helps your

yoga practice. By picking something you're passionate about, you'll be more likely to want to work on it, and you'll wrap it up sooner.

- If you have an unfinished project, list it anyway, and note that it's "still under development".
- Is there a "spin-off" you could do of your first project? For example, if your first project was a full-stack web app that let users find food inspection results for local restaurants, could you maybe also create a Chrome Extension that spoke with the same server?
- Are there non-product things you've done that could be worth listing? Have you volunteered at any coding events? Have you contributed documentation to an open-source project? Think back through the time you've been learning to code, and see if anything pops up that could be used.

Ultimately, a portfolio site is only useful when you have projects to highlight. If you don't have at least 1 significant project and 1 minor project, consider spending some time filling in your portfolio. This can be an unfair burden on folks with caregiver responsibilities, or any other life situation that limits their time or energy. If it's unworkable, you might have to shift strategies; there are many ways into the industry! Unfortunately, this book's strategy assumes that you either already have suitable side projects, or else have the resources to create them.

WHAT IF MY PROJECTS AREN'T GOOD ENOUGH?

It's very common for junior developers to feel like their work isn't impressive enough to get a job. You probably aren't super confident in your skills, and you might look at your projects and feel like they're nothing special / nowhere near as good as they could be.

This is all a *common and normal part* of entering the industry. Most of the senior developers who have successful careers started out exactly where you are; worried that they're not experienced or knowledgeable enough to get a job.

In fact, if you've spent many hours on a project, there's a very good chance that you've faced and overcome significant technical challenges. Projects often have *hidden complexity*, and

employers are eager to hire developers who are good at solving problems. The main purpose of a portfolio site is to show prospective employers how you ran into technical challenges, and overcame them. Your projects don't need to be flashy or feature-rich.

This is why a detailed portfolio is so important. LinkedIn lets you add a screenshot and a couple paragraphs about a project, but think about how incomplete that story is. Most junior-developer projects look the same on the surface; the interesting stuff is *below* the surface. And a portfolio site is a great chance to guide the reader through that interesting stuff.

Of course, there is such a thing as a project which isn't complex enough to warrant being put on a portfolio site. But folks tend to overestimate the level of complexity which impresses employers, when framed correctly. Remember, you're going for junior-level positions! Reasonable employers will understand that you're not an expert yet (and you probably don't want to work for unreasonable employers anyway, so nevermind what they think).

If you wait until you feel ready to start working as a developer, you'll never start!

If you identify as female, there's a good chance that you're underestimating your own skills[†], and overestimating the level required for these positions. If you're feeling like you're not as advanced as some of the men in your peer group (eg. fellow bootcamp alumni, members of the same "learning to code" community), it's worth considering if maybe your peers are overestimating their abilities.

[†] Source: <https://www.fastcompany.com/40554829/youll-never-guess-which-gender-tends-to-overestimate-their-own-stem-skills>

IN SUMMARY

- Development portfolios can be a **secret weapon**, precisely because very few aspiring developers put thought and energy into them.
- You want to highlight your best work in your portfolio. Between 2 and 5 projects is ideal.
- All sorts of projects are suitable for this section, but you should avoid:
 - Work created by following a tutorial or completing an exercise
 - Projects that aren't related to software development
 - Confidential work
 - The portfolio site itself, *unless* you can tell a compelling story about it that isn't just summarizing what the visitor has already seen

CONTINUE READING

You've reached the end of this quick preview. We've just scratched the surface. The good news is that the rest of the book is free!

You can sign up to receive the full book at this URL:

<https://joshwcomeau.com/effective-portfolio/>