Preamble
This is a short article describing my transition from data analyst to my first software engineer, and how I got there. I know I had a lot of concerns about being able to get a decent software engineering job, especially as someone who wouldn't likely be able to participate in Capstone, so hopefully this mess of words serves as some motivation or reassurance to you, as a Launch School student.

Disclaimer
All these views are mine, they're not endorsed by Launch School.

Launch School is not saying that you will get a job once you finish the backend portion of the course.

I am not saying that you will get a job/should get a job once you finish the backend portion of the Core curriculum. I don't want to set false expectations here.

Background
I'm a data analyst by trade. My interest in software engineering really started from my current workplace - I work for a SaaS company, and I was incredibly inspired by the software engineers' ability to translate an aspirational idea into a fully functioning thing that brought value to users and improved their lives. That inspiration coalesced into action on September 10th, 2020, when I signed up to Launch School.

On the 10th of August 2021, I was offered a Software Engineer position at my current company. I will be starting at the entry level of a 'Software Engineer' (I'm as junior an engineer as you can get, without being a 'junior software engineer', if that makes sense).

So how did you do it?
Are you super intelligent or efficient or something?
TL;DR - No.

As of writing this, I'm in LS202, or HTML/CSS, having recently finished the backend part of the course. I personally don't find discussions of "how long did you spend on X" very productive, but I'm cognisant that other people may find it useful.

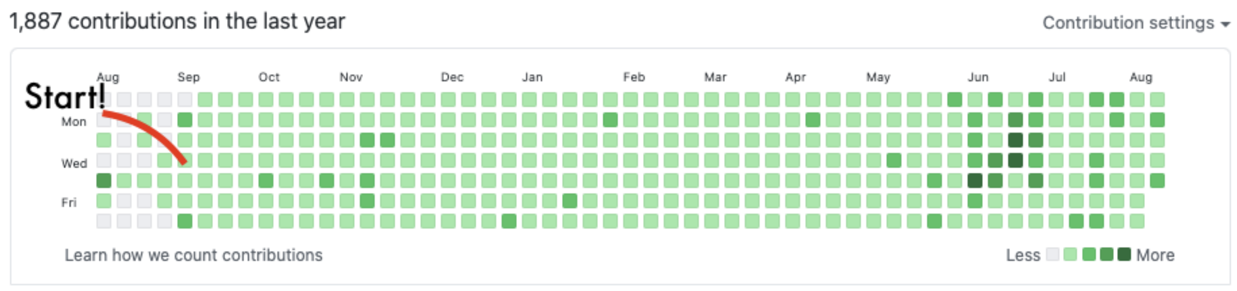
I've been enrolled at Launch School for 11 months. I have a full time job, so I study part time. An hour in the morning, an hour in the evening, and 6-8 hours over the weekend. That has resulted in 48,667 minutes worth of study thus far. That's a little over 811 hours of life. About 18% of that was RB109 assessment preparation. I've managed to get A+'s on all the assessments so far.

I have no idea if that is fast or slow, or good or bad. And I don't think it really matters. Everyone is different, and comparisons to someone who may have completely different circumstances to you probably isn't that applicable anyway.

I don't think I'm particularly gifted or intelligent (this is important later). I don't really have any specialised knowledge of software engineering beyond what I've done at Launch School. I'm a data analyst, so I like to think I'm pretty decent at SQL. I do *some* data engineering type stuff as part of work. But none of that is super relevant to software engineering, at least at my current level. I write a fair bit of code in R, but that's really just writing simple loops and using existing libraries to do data transformation. It's simpler than the code you'd write in RB109.

Progress is gated at Launch School. If you're passed the assignments, there's likely very little difference between our mastery of software engineering concepts.

I think if I had to compare myself to an average human being, I think the quality that I would compare favourably for would be persistence, perhaps to a fault. I have studied consistently since the 10th of September. I push my notes to a private repo on Github, so that serves as a good reminder of my 'streak'.



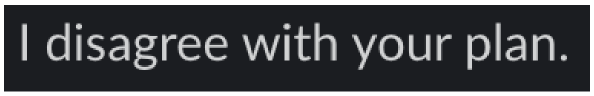
However, that's just my personal study habits. I like to study consistently over time. And I have the freedom to make sacrifices to achieve that study. Maybe you work better studying all at once over the weekend. Maybe your life situation is more demanding. Again, your circumstances are likely quite different to mine. In any case, if you're at Launch School, you're probably just as dedicated as me.

That's a lot of words to just say "If I can do it, you can too". That's probably cold comfort, particularly those of you who suffer a lot of imposter syndrome (I have a particularly bad case of this).

Anxiety & Misconceptions
But I don't have any real experience. What level will I be when I'm done with Core?
TL;DR - I haven't even done all of Core, but what I have done prepares you for non-junior engineering interviews.

I'd say probably the most influential thing that happened to me during my Launch School study was a frank chat I had with Chris. I spoke about some of my expectations as someone who'd finished the backend portion of Core, wanting to push for a role in software engineering.

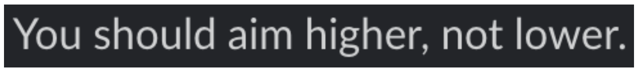
I have a pretty bad case of imposter syndrome, so I thought that only Capstone graduates were really ready for a 'real' software engineering job, and that having completed the backend portion of Core, I should be ready to start applying for junior engineering roles.


I disagree with your plan.

This might sound harsh out of context (sorry Chris). But this is exactly what I needed to hear at that moment. Chris writes about why he thinks this is generally a bad idea here (I had not read this article when I had this chat).

At that moment in time, I was starting to feel antsy (and probably experiencing some degree of burnout with my studies) about whether my studies would ever pay off. Launch School preaches the slow path, but I am naturally an impatient person. I am also cynical, and when I am told to 'trust the process', I think of a strange obsession about tanking (basketball reference). It has negative connotations to me.

I was panicking. That, combined with my imposter syndrome, made me believe that I should just settle for what I could get. I did not believe that Core would prepare me for a mid-level software engineering role.


You should aim higher, not lower.

This might sound trite. But it was very powerful for me. The cynic in me still thought I might not be good enough, but I felt that I owe it to myself to at least try.

I also think this statement is generally applicable to all Launch School students. Yes, you may feel significant imposter syndrome like me. You may not believe me, or think that I am peculiarly talented (I am not), or that I am lucky (I believe that there is always some degree of luck) or that you are somehow uniquely inferior. I am a sample size of one, but I think that if you can pass assessments at Launch School, you're probably equipped to interview for non-junior engineering roles.

I will caveat what I am saying above - while Launch School will equip you with the skills you need to pass coding challenges and that sort of thing, **getting to that stage in the first place is a whole different kettle of fish (which I think Capstone is more geared towards).**

How did you get your job?
How many applications did you do? Did you network/go to meetups? Would you write me a recommendation?
TL;DR - No. I focussed on smaller SaaS companies.

This is what worked for me, in a particular market, at a particular point in time, at a particular point in the business cycle. I make no guarantees about your success if you follow this approach. Your mileage may vary. Life is uncertain.

I ‘only’ applied to 8 jobs. I have the luxury of being employed in a job that pays the bills and isn’t injurious to my health.

Armed with some reassurance, I did not have to settle.

My approach was to focus on smaller, SaaS companies. I have experience working in a SaaS environment and a startup-ish environment. In my limited experience, these types of companies are more open to non-traditional backgrounds for software engineers, and more agnostic towards your software language background.

For example, I had one interview where they asked me about data structures and algorithms, things you’d learn about in a CS degree. I think my response was literally “I have no idea about any of this stuff”. The HR person simply laughed and said “Ok that’s fine, I’ll send you the coding challenge anyway”.

They sent me a coding challenge, saying they wanted me to write it in Java. Again, I said “I don’t know Java, can I do it in Ruby?”. Again, they laughed and said “Yeah that’s fine”. I got positive feedback from that coding challenge, but ended up declining going any further in the interview process due to getting an offer from my current company. I think that a good software engineering environment probably won’t mind what language you’ve learned, so long as you can demonstrate good proficiency with the fundamentals of the language.

Because I’m a slight masochist, of the 8 applications, I applied for 3 junior roles, all at larger, more established, non-SaaS companies. I got ghosted by 1 of them. 2 responded with generic rejection emails. I pestered the companies with the generic rejection emails for feedback (I told you I am persistent). The gist of the responses was “Sorry, we’re looking for new CS graduates with demonstrated experience in Java/C++/.Net”. Ignoring the oxymoron in that statement, while these are shamefully small sample sizes, these experiences seem to affirm my flimsy hypothesis that you’ll have better luck at SaaS companies.

Turns out my current company is hiring currently. I reached out to the principal engineer. He is a nice person. I did the coding challenge. I did reasonably well.

That’s a fluke! You’re just lucky!
Yes. I was in the right place, at the right time. But I was equipped to take advantage of that situation because what I’ve done in Core prepared me for it.

How did you perform on the assessments/coding challenges?
I’ll say again that I don’t have any specialist knowledge of software engineering that would give me an edge over you, if you’ve completed the backend portion of the course.

All told, I had to complete three different coding challenges across the different companies I applied for.

They were broadly speaking, all of a similar difficulty and approach. I had to read an external file (e.g. a .csv, or .json file) into an application, do some sort of processing to 'clean up' the data so that it fit into structure, such that the requirements of the challenge were met (e.g. the application needs to calculate an average of some values).

All of these challenges had some form of 'gotcha' in them - e.g. the input files would be named inconsistently, or they were of different formats or structure. I had to write tests for these use cases, with not much guidance beyond that.

After having completed the backend portion of Core, none of these tasks were particularly difficult, just time consuming. The feedback was generally quite positive, commenting on how I was able to:

- deconstruct the problem into smaller, manageable chunks
- identify edge cases
- write tests (although in general, all of my interviewers across different companies disliked Minitest and recommended I use RSpec)
- reason about how my application should be structured in terms of classes and responsibilities
- what data structures I could/should use
- talk about trade-offs in an intelligent manner
- communicate these in an accurate and succinct manner in a README.

All of this is just RB109, RB120 and RB139, at different levels of abstraction.

So yes, you'll do fine once you get to interviews and coding assessments, even before you completely finish Core. What I've experienced of Core will get you across the finish line. You just have to get to the start line.

So what now?
Apparently I am quite useful in my current role, so I am not starting the engineering role for 6 weeks until I hire my replacement and onboard them.

I feel vindicated, terrified, excited and uncertain. I have no idea how well I will do as a 'real' software engineer. But I can only do my best, and I think I am well equipped to do the job.

Oh, and I'm still planning to continue with Launch School. I like the style of instruction and the self-styled pacing. There's far too much useful knowledge left in the Core curriculum for me to stop now.