Should I Open Source my Company?

2022-03-25 • 11 minute read

Supabase has been open source from day one. This is one of the best decisions we've made. It didn't feel like a difficult decision at the time, since kiwicopple and I are long time users and advocates of great open projects like PostgreSQL, Python, Bitcoin, React, [... insert long list]. Open-sourcing Supabase ended up surprising us in many ways. Many people imagine that maintaining your business in public might be burdensome - but the opposite is true. There are many unexpected upsides that have made building Supabase - the product and the company - easier.

While some of this advice comes from our lens as a Dev Tools or PaaS company, most of it will apply to any software company.

When we discuss open source business models with other founders, there are three complaints that come up again and again. These are:

- People might criticize my messy/bad/ur
- Hackers will find and exploit security ho
- Competitors will steal my Intellectual P

We'll cover each of these in detail and follow up with some of the surprising benefits we discovered building Supabase, namely how open source:

- Solves your dev talent problem
- Improves your product (massively)
- Will carry us to Utopia

Let's dive in.

But my code is bad

This is just ego. The person who spends the most time thinking about you is you, and the person who spends the most time stressing over your bad code is you as well. If someone else is spending time stressing over your bad or messy code, then it's likely you've hit upon something worth pursuing. The folk story of the Stone Soup applies here. If you share good ideas with the community they will adopt and improve on them with you. The community will help refactor and replace any bad code, and they might even introduce new code quality guidelines to your project, improving the process for everyone who comes after. Toxic community members who complain about bad code instead of making suggestions to improve it are not the people you want in your community anyway.

All open source contributors have a graveyard of old projects. What's cool about this is that people looking to build something similar have a library of ideas and approaches to learn from. They may take inspiration from your approach to certain problems. At the very least, they won't be wasting time making your same mistakes. Don't worry about old projects ruining your reputation. At best people will find them useful. At worse they won't get noticed anyway.

Getting to secure

One of the biggest fears running a hosted solution where the code is publicly available is malicious actors will read your code, discover exploits, and hack your services. In our experience, Linus's law of "given enough eyeballs, all bugs are shallow" also applies to security issues. Since the very early days of Supabase we have had (often anonymous) security researchers active in our code and platform, helping us find potential issues. If you're offering a so

instructions available at /.well-known/secu

disclose potential issues to you.

The premier example of how open source projects can be more secure than proprietary code bases is Bitcoin. In his 2015 talk Andreas M. Antonopoulos describes how closed source banking systems have the software equivalent of weak immune systems, because huge security holes can be obfuscated for long periods of time, and when eventually exploited can have enormous detrimental effects. On the flip side of this is an open source protocol like Bitcoin, where any security holes are there for all to see. Exploits are found early and often, and then patched. Remember that successful software companies can take more than a decade to build. Over a long time period, open source systems will tend towards a more secure state over secretive, proprietary systems, especially in a time when hiring talented security engineers is extremely competitive.

Beating the competition

In software ideas are cheap. Value is almo
When you share your ideas with the world
that matter most. Instead of spending tim

partners have access to your code base, you can focus on iterating faster. You'll probably be surprised how the open source community responds to your openness.

Grow a community of experts

In order to build a successful company, you'll have to execute well and grow your customer base over a long period of time, let's say 10 years. In this Ultra Marathon, any advantages gained by competitors peeking at your code base are unlikely to be material over that 10 year period. It's likely that you'll have to iterate on, and refactor your code base many times over during that period. Building a team and a community that is the best in the world at iterating on *your* solution is how you will win in the long term.

Winning in the market

If you're in a market with a huge (and growing) TAM, that is *really* worth pursuing, it's unlikely to be a winner takes all situation. The first lesson is to find a segment of users you can "wow" and then iterate like crazy. Every second you focus on your competitors is time you could be focused on improving your product or your team. If a competitor forks your open source project, and is able to out-ship you, then I would argue that you were bound to lose that particular race anyway.

On top of all this, IP and patents in software are notoriously difficult to enforce. It will save you time, money, and mental energy if you're not worrying about trying to protect your software ideas with lawyers. You can protect your business much better by becoming the best in the world at implementing your software ideas.

Late stage

Once your project reaches significant scale, you might find yourself in a situation like Elastic, or Mongo, where large cloud providers are offering your product with a superior distribution model.

Firstly, if you're just starting out, the bigge

to that point. If you become so successful away from one of it's many billion dollar product lines to compete with you in hosting your product, you're doing something *very* right.

Secondly, and more constructively, you should prepare for this eventuality by finding areas where you can outcompete anyone. Most cloud providers are notoriously bad at Developer Experience for example, so take advantage of that and make DX one of your core competencies. If after 6 years, Google tries to steal your lunch, you should have a brand, a team, and a community, that have spent the last few years preparing for a David versus Goliath-type fight. Make sure you're not blindsided by something like this by planning for it from the beginning. You have enough time and focus on your side to construct a winning strategy.

Stop worrying

In general if a competitor is looking to you for ideas, then good luck to them. They'll always be a step behind you, and expending more energy than you are by worrying about their competition. Especially in the early stages, when resource is scarce.

Now that we've addressed some of the common objections, let's move on to some of the surprising benefits.

Hiring developers

One of the biggest complaints of startups - no matter the size - is how hard it is to hire developers. Sourcing is hard, assessing developers is hard, convincing them of the quality of their future colleagues is hard. Being open source can solve your developer hiring problems!

Sourcing openly

All developers benefit from open-source, a
benefits forward by contributing to open s
they contribute because of a good commu
technology, or enjoy solving hard technica

you lower the barriers to contribute code to your open source project, there's a good chance great developers will find your project. They might contribute in issue discussions, solve bugs, identify problems, or review PRs. Every contributor may not be open to work, but many will be attracted by the prospect of working on open source code as part of their full time gig. It's a huge advantage open source companies have over closed sourced ones. At Supabase we didn't start doing outbound recruitment until after we already hired 32 developers, and been operating for more than 2 years. Open source has been, and will continue to be, our primary source of finding new (and incredible) talent. Being a remote and async company, we find having an open source hiring strategy means candidates are self selecting for these skills, which makes our job of screening for them much easier.

No, I won't take home your take home test

One of the biggest complaints developers make while looking for their next gig is having to solve LeetCode problems or do take-home tests in the interview process. They are time-consuming, arduous, and often don't represent the type of work they'll be doing on the job. However, if someone contributes to your repo you already have:

- examples of how the person communicates with team/community members asynchronously and in a remote setting.
- examples of *real world* code contribution.

And the candidate gets to see all this from the company side as well! Does this team value code quality? Do they communicate effectively? Do they make good technical decisions?

People who have been hired at Supabase through our repos have made themselves indispensable. Many of them drove the development of existing repos, and even started new client libraries and SDKs. And not one of them were ever required to solve a LeetCode problem over Zoom.

But does it integrate with Exc

When building a startup, you might find th

your biggest problems. In order to move fa user base is asking for. The features requested by the minority are prone to getting indefinitely shunted. When your systems are open source however, these users have the ability to contribute the features themselves, expanding the usefulness of your software - sometimes to new use cases that you didn't even know existed. This happens most often with integrations and adaptors. Users might want an Azure or Vercel integration, and if someone else can provide the development resources to make those connections, then everyone using your project benefits as a result.

It's 2022. Where are the flying cars?

In a world with no open source, technology companies everywhere are re-inventing the wheel, on loop... forever. To combat this at Supabase, we don't just open source our new projects. We always try to support existing open source projects, before inventing new ones.

Think about each of the automotive companies today. Writing proprietary functions, to build proprietary UI elements, on proprietary windowing libraries, running inside proprietary operating systems, using proprietary drivers, to display an error message that you'll probably never see! What a waste of developer time and effort!

I won't evangelize too much here... I think this meme summarizes my feelings on the subject...

More evidence please

Next week is Supabase Launch Week. It's the culmination of everything the team and community have spent the last 3 months working on. If you're not convinced by my arguments above, I believe the rate at which the Supabase community is progressing, in such a short time, speaks to all the benefits of running an open source company. See the Launch Week 4 post for all the details. And visit our GitHub if you want to get involved.

Share this article

Last post

Supabase Launch Week 4
25 March 2022

Next post

Postgres Auditing in 150 lines of SQL
8 March 2022

Related articles

What is SAML? A practical guide to the authentication protocol
Brace yourself, IPv6 is coming
Using React Query with Next.js App Router and Supabase Cache Helpers
Elixir clustering using Postgres
Supabase Beta December 2023

View all posts