

// phase 6 · the final phase

Launch & Signal



Building the project was only half the work. This is the other half: getting the right people to see it, and turning that attention into interviews.

baswe.ai // become a software engineer

This is the in-depth guide to phase six of the Signal Project, the final phase. Work through it once you have a shipped MVP and the raw material from phase five. By the end you will have launched your project like a startup and started turning that attention into interviews.

Building it was only half the work

Here is the truth almost no one acts on: a great project that nobody sees does nothing for you. Building is fifty percent of the work. Getting the right people to see it is the other fifty percent. Most engineers build something decent, quietly add it to a resume, and then apply into the void, and then they wonder why nothing happens. You are going to do the opposite. You are going to launch this like a startup and make sure the right people see it.

Reframe what you are doing. You are not posting a project. You are launching a product and running a small distribution campaign. Every star, every user, every share, and every person who reads your write-up is social proof that you build things that matter. That proof is what turns hiring you into an easy yes.

Even without a real startup, the more publicity, the more stars, and the more people who use or want your tool, the more clout you have, and the easier it is to get hired. Attention is the asset. Go get it.

What launch actually means here

It is not one post and done. It is a sustained push across several channels over days and weeks, running in parallel with direct outreach to the specific people who can hire you. Two tracks at the same time: broad publicity to get seen and build proof, and direct outreach to get in front of hiring managers.

The goal is not to go viral. It is to gather enough real signal, such as a few hundred GitHub stars, a few hundred to a thousand article reads, some real users, and some genuine conversations, to prove the project matters, and to get it directly in front of the people who decide whether you get hired.

Before you launch: get the assets ready

You only launch once, so set the assets up first. The README and the article matter most, because companies click the repo and read the story.

- ▶ A clean public GitHub repo with a strong README: the problem, your approach, screenshots or a demo, the architecture diagram, and how to run it.
- ▶ A live demo or a short demo video, so people can see it work without installing anything.
- ▶ A simple landing or product page (optional, but it makes the project feel real).
- ▶ A written story: a Medium or blog article walking through the problem, your research, your decisions, and what you built.

- Your design doc and decision log, ready to share. These are your differentiators.
- A short demo clip, a few screenshots, and a one-line and one-paragraph description you can reuse everywhere.

The GitHub play

GitHub is where companies verify that you can actually build. Make the repo excellent, then earn stars as social proof.

▪ Make the repo excellent

The README should sell the problem and the solution: a clear headline, the problem, a demo clip or screenshots, the key features, the architecture diagram, how to run it, and the tech used. Keep the commit history clean, the structure clear, and add a license. Someone should understand what your project is and why it matters in about sixty seconds. Then pin it on your GitHub profile and make the profile itself look like an engineer's, with a real bio and your best work pinned.

▪ Earn stars

Stars are a simple, visible signal that the project matters. Earn them by getting real eyes on the repo: share it everywhere you launch, ask your community, your teammates, and your network to check it out and star it if they find it useful, and submit it to relevant directories and "awesome" lists in your space. A few hundred stars is strong proof. Do not fake engagement. Earn it by getting the right people to actually look.

The LinkedIn strategy

▪ Post the launch, then keep posting

Announce the project with the story, not just "I built X." Lead with the problem and why it matters, then what you built, then the result and a demo, then the links to the repo and the article. Always include a demo clip or screenshots, because visual posts get far more attention. Then keep posting. Share the article, a decision you made, a milestone like passing a star count or getting your first users, and a lesson you learned. From the moment you launch until you are hired, keep the project visible. Every post is another chance for the right person to see it.

▪ Reach out to hiring managers directly

This is where launch turns into hiring. Use your silent network from phase two and a little new research to find the right people: the hiring manager or engineering lead for the team you would join, recruiters, and the connections you already made. Then message them, and lead with the work, not the ask. Do not send "please hire me." Send the actual project.

Crucially, send them your docs. A resume looks like every other resume. A working product, a clear write-up, and a real design document with your decisions and

tradeoffs make you look like a senior engineer, and almost no other applicant does this. Link the repo, the demo, and the design doc. Use the template below.

HIRING-MANAGER OUTREACH TEMPLATE (LEAD WITH THE WORK, NOT THE ASK)

Open. Reference the connection, or something specific about their team, their company, or a problem they have.

The work. "I built [project] to tackle [the problem your company has]. Here's the repo, a short demo, and a write-up of the engineering decisions behind it: [links]."

The proof. "It's picked up [stars / users / feedback] since I launched it."

The ask. "I'd love your feedback, and if your team is hiring for something like this, I'd be glad to talk." Or ask who to contact, or for a referral.

Rules. Short, specific, and genuine. Send the actual work. Never lead with your resume.

For a warm path, reconnect with the people who responded during your research: tell them you shipped it, and ask if their team is hiring or if they would refer you. For a cold path, find the person on LinkedIn, get their email with a tool like hunter.io, and send that same short, specific message with the links.

The Reddit and community play

▪ Find the right communities

Identify the subreddits, Discords, Slacks, forums, and newsletters where your target users and your industry actually spend time. Do not stop at the generic programming subreddits. The industry-specific and problem-specific communities are where the people who care about your problem, and the companies that hire for it, actually are.

▪ Contribute first, then share

Do not just drop a link, because you will get removed or downvoted. Participate first. Answer questions related to your problem, be genuinely helpful, and then share your project where it is relevant and allowed, offer it for free, and ask for feedback. Answering questions about your problem positions you as someone who understands it, and naturally surfaces your project. Communities reward real contribution and punish drive-by promotion.

▪ Where else to post

Post where your users and target companies are: Hacker News with a Show HN if it fits, Product Hunt, Indie Hackers, dev.to, relevant Discords and Slacks, and X. Repurpose the launch for each place rather than pasting the same thing everywhere.

The content play

▪ Write the story article

Write a Medium or blog article that walks through the problem and why it matters, the research you did, the key engineering decisions and their tradeoffs, what you built, and what you learned. This is the piece that makes you look like an engineer who thinks, and companies love it. Publish it, then share it everywhere, more than once. Your design doc and decision log basically write this article for you.

▪ Make many pieces from one project

You built one thing, so make many pieces of content from it: the main article, a technical deep-dive on a single decision, a short "what I learned" post, a demo video, an X thread, a LinkedIn carousel, and an architecture breakdown. Each one is another surface for the right person to find you. Repurpose relentlessly, because the same project can generate weeks of content.

Turn attention into interviews

Publicity is the means. A job is the end. As the project gains traction, convert it deliberately.

- ▶ Reconnect with your network: "I shipped [project]. I'm starting to look at roles. Is your team hiring, and would you refer me?"
- ▶ Apply to your target companies, but never just apply into the void. Pair every application with direct outreach and your project as proof.
- ▶ Use the social proof in the conversation: point to the stars, the users, and the feedback. Proof that people care about what you build is proof that you will deliver value.
- ▶ In interviews, your design doc and decision log are your script. You can walk through the problem, the decisions, and the tradeoffs like a senior engineer.

The resume gets you screened out or ignored. The project, the proof, and the direct outreach get you hired. Lead with the work, not the resume.

Keep the momentum until you are hired

Launch is not a day, it is a campaign that runs until you have an offer. Keep posting, keep sharing, keep reaching out, and keep improving the project based on the feedback you get. Every week the project stays visible is another week the right person might see it. The people who win here are not the ones with the flashiest project. They are the ones who kept showing it to the right people and did not go quiet.

Launch checklist

Hit all of these. Treat it like a campaign, not a single post.

LAUNCH CHECKLIST (HIT ALL OF THESE)

- [] Public GitHub repo with a strong README, demo, and architecture diagram
- [] Live demo or demo video
- [] Medium or blog article telling the story
- [] LinkedIn launch post with a demo clip, then keep posting
- [] Reddit and industry communities (contribute first, then share)
- [] X, Product Hunt, Hacker News, Indie Hackers, as relevant
- [] Direct outreach to hiring managers and your network, with links and your design doc
- [] Repurposed content: threads, carousels, deep-dives
- [] Applications paired with outreach and proof
- [] Ongoing posting until you are hired

Common mistakes to avoid

- ▶ **Building something great and never launching it.** The most common and most costly mistake. The work is only half done until it is seen.
- ▶ **One post and done.** Launch is a campaign, not an event. Keep going until you are hired.
- ▶ **Just adding it to a resume and applying into the void.** Pair every application with outreach and proof.
- ▶ **Dropping links in communities without contributing.** That is spam and it gets removed. Add value first.
- ▶ **Leading with "please hire me."** Lead with the problem and the work. The ask comes last.
- ▶ **Sending a resume when you could send a working product and a design doc.** The work is far more convincing.
- ▶ **A messy repo with no README.** Companies bounce in seconds. Make it excellent.
- ▶ **Faking stars or engagement.** Earn real attention. Faked proof is worse than none.
- ▶ **Going quiet after launch.** Sustain the campaign. Visibility over time is what converts.

What a successful launch looks like

By the end of this phase, you should have:

- ▶ The project launched across GitHub, LinkedIn, Reddit and communities, and a written article.
- ▶ Real social proof: stars, article reads, some users or feedback, and some conversations started.
- ▶ Direct outreach sent to hiring managers and your network at target companies, with your project and design doc as proof.

- Applications paired with outreach, never sent into the void.
- Ongoing posting keeping the project visible.
- The right people have seen it, and it is turning into conversations and interviews.

That is the whole Signal Project

You picked a target, researched it, chose a real problem, became an expert on it, designed a system, built it with a team, and now you launch it and put it in front of the people who can hire you. Most engineers never do any of this. You did all of it. The project proves you solve real problems. The launch makes sure the right people know it. Now go get hired.

→ You've done the work. Now launch it loud, get it in front of the right people, and go get hired.