Table of Contents

The First Third: Show Up, Set Up, Shut Up

Make Your Goal Painfully Clear

Meet Everyone (Yes, Everyone)

Map the Brain Trust

Build a Roadmap You Can Actually Finish

The Middle Third: Build the Thing

Deepen Relationships Past the Small Talk

Deliver the Core Functionality

The Check-In Loop

The Final Third: Make It Real

Operationalize the Project

Polish, Extend, Document

Make Your Case (Without Making It Weird)

Final Thoughts

How to Lock In a Return Offer as an Intern

Internships at big tech companies are a strange beast. You are simultaneously expected to deliver production grade work and to be the new kid who doesn't know where the bathroom is. Worse, you have roughly 10 to 12 weeks to convince a group of strangers that they should pay you a six figure salary to come back next year. No pressure.

Why can I teach you this? I've gotten return offers at 3x internships (small startup, Lockheed Martin, and Amazon, where I returned fulltime) and mentored multiple interns while I was at Amazon who all received return offers. This guide is based on that experience & updated according to my current contacts across multiple big tech companies.

The good news is that getting a return offer is not actually about being the smartest engineer in the building. If that were the bar, almost no intern would ever get one, because you're surrounded by people with 5, 10, 20 years more experience than you. Return offers go to interns who:

- Are trusted to ship work without a babysitter
- Communicate clearly and often
- Build relationships across the team and beyond
- Finish what they start

That's it. I'm going to break the internship into three phases (first third, middle third, final third) and walk through what you should actually be doing during each one. Some of this will feel obvious to you, but most people STILL won't execute on it. Be one of the people who does execute.

A quick note before we get into it: this guide is aimed at internships at larger companies (think FAANG, unicorns, big public tech companies). At a 10 person startup, the dynamics are different and frankly easier in some ways since you're working directly with leadership every day. The advice still mostly applies, just scaled down.

The First Third

The first third of your internship (roughly weeks 1 through 4 of a 12 week internship) is not about coding. I'll say that again because one of the biggest mistakes I see interns make: **the first third of your internship is not about coding.**

You will be tempted to open your laptop on day one, find the first ticket with your name on it, and start cranking. Resist that urge. Your focus here is to build relationships lay the groundwork for a successful second third.

Make Your Goal Painfully Clear

In your very first 1:1 with your manager, tell them you want a return offer.

This sounds ridiculous to a lot of people, but here's the thing: your manager is busy. They have a full time team to run, KPIs to hit, performance reviews to write, and now they have you, a temporary employee who shows up for three months and then vanishes. If you don't tell them what you want, they will assume you're "just trying it out" or "still figuring things out." That is the worst position to be in.

What this sounds like in practice:

"Hey, I just want to be upfront about what I'm hoping to get out of this summer. My number one goal is to come back full time, so I'd love to know what 'exceeds expectations' looks like for an intern on this team. What are the specific things you'd want to see from me by the end of the summer for that to happen?"

Now your manager has a clear ask, and more importantly, they have to give you criteria. Write down whatever they say. Bring it back up in future 1:1s. This becomes your rubric for the entire summer.

Two things worth knowing about how return offers actually get decided (it’s quite similar to how promotions work):

1. At most big companies, your manager submits a recommendation, but the final call often involves a calibration meeting with other managers and skip-levels. Your manager (and sometimes your team) is essentially your advocate in a room you'll never be in. Give them ammunition.
2. The decision is often made earlier than you think. By week 8 or 9, opinions are mostly formed. By the time you give your final presentation, the decision is already 90% baked. This is why the first third matters so much.

Meet Everyone

Set up 1:1 coffee chats with every single person on your team in the first two weeks. Not just your buddy or your mentor. Everyone. The senior engineer influences the team, the PM, the TPM, the designer if you have one, the staff engineer who works on something tangentially related, etc.

Then go one ring further out. Ask your manager: *"Who else outside the team should I meet? Are there people on adjacent teams I'll be working with, stakeholders in my project, or folks whose work touches what I'm doing?"* Set up chats with those people too. A good manager might do this for you without you asking, but it’s not guaranteed.

Keep the chats short (20 minutes is plenty) and have a loose script:

- Ask a little bit about their personal life (hobbies, places in town they like, etc)

- Tell them a bit about you and what you're working on
- Ask them about their work and how it connects to yours
- Ask what advice they have for interns who've done well on this team
- Ask if there's anyone else they think you should meet

Why does this matter for a return offer? Two reasons. First, when calibration happens, the people who know you will speak up. The people who don't, won't. Second, and this is the deeper reason, the people you meet will end up being the people you ask for help from later. And asking for help from someone you've already had coffee with is roughly 100x easier than cold-DMing a senior engineer you've never spoken to.

Map the Brain Trust

As you meet people, build a mental (or actual, no shame in a doc) map of who knows what. This is one of the most underrated skills in any large engineering org, and you should start practicing it from day one.

In any team of more than 5 people, knowledge is heavily distributed. There's almost always:

- The person who actually knows how the deployment pipeline works
- The person who wrote the gnarly legacy module everyone is afraid of
- The person who knows the historical context for why X decision was made
- The person who has the best taste for code review
- The person who has the most political capital with leadership

These are usually not the same person. Figuring out who to ask for what is a superpower. It also makes you dramatically more efficient, because you're not bouncing your questions off the wrong people and wasting their time (or yours).

A practical tip: when someone helps you solve a problem, take 30 seconds to note what they helped you with. Over a few weeks, you'll have a reference of "go to X for infra questions, Y for product context, Z for code review on the auth flow."

Build a REALISTIC Roadmap

By the end of the first third (or sometimes by the end of week 2), you should have a written project plan. Not a vague "I'll work on the X service" sort of thing. An actual, week by week breakdown of what you're going to deliver and when.

Here's the structure I'd recommend:

- **Final deliverable:** What does "done" look like? Write it in one sentence. If you can't, you don't understand the project yet.

- **Milestones:** Break it into 3 to 5 chunks. Each chunk should be something demoable. Critical these demos go well and are easy to understand so you can show incremental progress.
- **Stretch goals:** What would you do if you finished early? (You should aim to finish 1-2 of these, but don't declare this upfront)
- **Risks and unknowns:** What could blow up your timeline? Managers love this section because it's what they actually worry about. It's also quite hard to scope when you lack context. Lean on your team for help with this.
- **Dependencies:** Who do you need things from? When do you need them by? These can easily blow up your timeline if not managed properly. Have a plan to go around these if you can't get them complete.

Then, and this is the critical part, **share it with your manager (and team) and get buy-in.** Don't just send it over Slack and hope they read it. Walk them through it in a 1:1. Ask: *"Does this look reasonable? Anything I'm missing? Is the scope right?" (pro tip: ask a trusted coworker to review it with you before you take it to your manager)* Some companies have a formalized doc review process for this, but if they don't, you should do it anyway.

Two reasons this matters:

1. If your scope is wrong, now is the time to find out and correct it so you can deliver enough to get the return offer.
2. You have just created a contract which you can benchmark to get your return offer. When you deliver against this plan, you have evidence. When something slips, you can have an honest conversation about why, rather than your manager just thinking "huh, the intern is behind."

I cannot stress enough how important it is to set realistic scope. The single most common failure mode for interns is biting off too much, getting halfway through, and ending the summer with a half-finished project that nobody can use. A small project, fully delivered and operationalized, beats an ambitious project that's 60% done.

The Middle Third

The middle third (roughly weeks 5 through 8) is when you actually do the work. This is the part you've been preparing for. By now you should know who's who, what you're building, and what success looks like. Now you ship.

Deepen Relationships Past the Small Talk

The relationships you started building in the first third need to evolve from "we had coffee once" to "this person genuinely knows me and would advocate for me." This happens through consistent, low-stakes interactions.

Some practical ways to do this:

- **Be present in team rituals.** Show up to standup with your camera on. Stay on the team Slack channel. Make jokes when appropriate.
- **Help other people.** This is huge. If a teammate has a question in a channel and you know the answer (or can find it), jump in. Interns who only ever consume help and never give it stand out badly. Interns who occasionally offer help stand out in a great way.
- **Show up to social stuff.** Team lunches, happy hours, offsites, whatever your team does. You don't have to be the life of the party. Just be there. Some of the most important impressions of you will form over a beer or a slice of pizza, not in code review.
- **Don’t just hang out with other interns.** I see SO MANY interns just spending time with other interns in their class at lunch / after work, etc. This is fine, but those are NOT the people who can actually influence your return offer. Befriend your intern class, but prioritize interacting with your team over interaction with them when you have the choice.

A note for the introverts: you don't need to become a different person. You just need to be visible. Sitting with the team at lunch and 5 minutes of casual chitchat in standup is plenty.

Deliver the Core Functionality

Your project plan said you'd ship X by week Y. This is when you make that happen.

A few principles that will make your life dramatically easier:

Ship small and ship often. Don't disappear for two weeks and then drop a 4000 line PR (I’ve seen SO many interns do this). Break your work into the smallest possible reviewable chunks. Open draft PRs early. Ask for feedback on the design (system design hehe) before you write the code.

Write code that looks like your team's code. Read the existing codebase. Match the style. Use the same patterns. This is not the time to introduce your hot take on dependency injection. Boring, conventional code that matches the team is way more impressive than clever code that doesn't. Use existing methods and already written code whenever possible.

Ask questions, but ask them well. There's a huge difference between "I'm stuck, halp" and "I'm trying to do X, I tried Y and Z, here's the error, I think it's because of A, do you have a sec to look?" The second version shows you've put in the work. The first version makes you look helpless. The general rule I'd give: spend at least 45 minutes trying to figure it out yourself, then ask. 3 hours is too long, 3 minutes is too short.

Don't hide problems. If you're behind, tell your manager you're behind. If something is broken, say so. The worst possible move is to pretend things are fine until they very obviously aren't. Managers respect honesty about problems way more than they respect heroic last minute saves (which usually involve someone else cleaning up your mess).

Check-In Loop

By the middle third, you should have an established rhythm of check-ins. At minimum:

- **Weekly 1:1 with your manager.** Come prepared with: what you shipped, what you're working on, what you're blocked on, what you need. Don't make them drag this out of you.
- **Bi-weekly check-ins with your mentor/buddy.** Quick, informal. "Here's what I did yesterday, here's what I'm doing today, here's where I'm stuck." This can and should literally just be a Slack message.
- **A midpoint review.** Most internship programs have this formally built in. If yours doesn't, create one. Around week 6, ask your manager: *"How am I tracking against what we talked about in week 1? What should I be doing differently in the back half?"*

That midpoint conversation can be the most valuable conversation of your summer. At this point, you can still course correct if you're behind. After this, your manager's impression of you starts to harden. Use it.

What you want to hear in that midpoint review is either "you're tracking well" or "here are 2 to 3 specific things you need to improve." If you hear vague stuff like "yeah, things are going fine," push for specifics. *"That's great to hear. If you had to name one or two things I could improve in the back half of the summer, what would they be?"*

Sometimes the feedback will sting. Don't get defensive. Say thank you, write it down, and actually act on it. The interns who get return offers are almost always the ones who visibly incorporate feedback. Your manager mentioned in week 6 that your PRs are too big? In week 7, your PRs are smaller. They notice. I promise they notice.

The Final Third

The final third (roughly weeks 9 through 12) is where return offers are cemented. By now, the core of your project should be working. The temptation at this point is to coast, polish the demo, and start thinking about your final presentation. Don't.

The final third is when you turn a cool side project into something the team will actually use.

Operationalize the Project

A working prototype is not a finished project. A finished project is one that:

- Has tests (unit & integration)
- Has monitoring and alerting if it's something that runs in production
- Has a deployment pipeline that doesn't require you specifically to run a magic command

- Has been reviewed and approved by the people who will own it after you leave
- Will not break on the first weird edge case it encounters in production

I've seen interns build genuinely impressive things that never shipped because they were 80% done at the end of the summer. The thing got demoed, everyone clapped, the intern left, and then the project sat in a branch forever because nobody else wanted to finish it (this happened to my first intern project, and I managed to get the return offer, but it's generally a negative signal). That intern's manager remembers them as "the one whose project we never ended up using."

Concretely, in weeks 9 and 10, you should be asking yourself:

- What needs to happen for this to run without me?
- Who is going to own this when I'm gone?
- What edge cases haven't I handled?
- What happens when this breaks in the middle of the night?

Then build for those answers.

Polish, Extend, Document

Once the core is solid and operationalized, *now* you can add the nice-to-haves from your stretch goals list. But only now. Don't get distracted by shiny features when the core isn't done.

Documentation is the unsexy hero of the final third. Write:

- A README that someone new could read in 10 minutes and understand what your project does
- An architecture doc explaining the major design decisions (especially the ones that weren't obvious)
- Runbooks for common operational tasks ("how to deploy", "how to debug X", "what to do when Y alert fires")
- A handoff doc for whoever is taking this over

Good documentation does two things for your return offer chances. First, it shows that you think about the long term and about people other than yourself. Second, it leaves a permanent artifact of your work that lives on after you do. Six months from now, when someone on the team is using your runbook, your name is at the top of it.

Make Your Case (Without Making It Weird)

In your last two or three weeks, you need to do a final round of relationship work. Specifically:

- **Bring it up with your manager again.** "Hey, I know we talked about return offers at the start of the summer. I'd love to check in on where I stand and what the timeline looks like for that decision."
- **Get feedback from the people you've worked with.** Not for the offer pitch specifically, but genuinely, ask for feedback. "What's something I did well this summer? What's something I could have done better?" Your manager is going to ask them these questions for your final evaluation. Get ahead of them and plant the seeds for what you want them to say.
- **Make sure your work is visible.** Your final presentation is a chance to show what you did to people who weren't in the weeds with you. Practice it. Get someone to give you feedback on it. Don't wing this. A great final presentation can slightly tip a borderline decision.

A small but real thing: at most big companies, there's a window where extending an offer requires headcount approval, and the team might know whether they have a slot for you well before the official decision comes through. If you've built a real relationship with your manager, they may give you a quiet heads up. Don't push for it, but don't be surprised if it happens.

Final Thoughts

Getting a return offer isn't about being a genius. It's about being someone the team trusts to come back. That trust gets built through hundreds of small interactions over the summer: showing up, communicating, finishing what you start, being honest about what you don't know, and being someone people enjoy working with.

A few last things worth saying:

- If you don't get the offer, it's not the end of the world. Plenty of great engineers didn't get return offers from their first internship. Take the feedback, learn from it, and apply somewhere else with a stronger story.
- If you do get the offer, you don't have to take it. The leverage you have with a return offer in hand for full time recruiting is enormous. Use it.
- The relationships you build this summer will outlast the job itself. The people you work with as an intern will end up scattered across the industry over the next decade. Be the kind of intern they want to refer. People from my intern class at Amazon are now distributed across DataBricks, Netflix, YC-Backed startups, Meta, etc.

Good luck.

P.S.
If you want more advice directly from me, check out my app and discord community The Daily Dev.
It's like Duolingo for System Design, and you get to join a discord community of engineers practicing the same skills as you for the same interviews and promotions. Available on web and IOS.