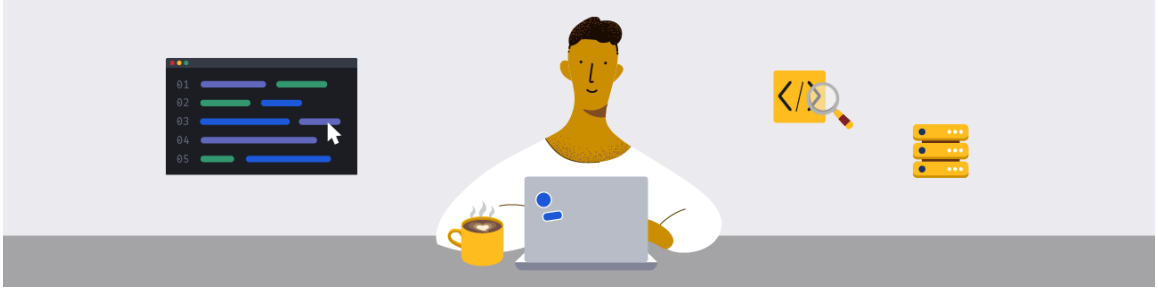
Mattermost

The Ultimate Guide to Building Technical Teams

How to Hire, Retain, and Nurture Great Engineering Orgs



Whatever your business, your developers are your biggest asset

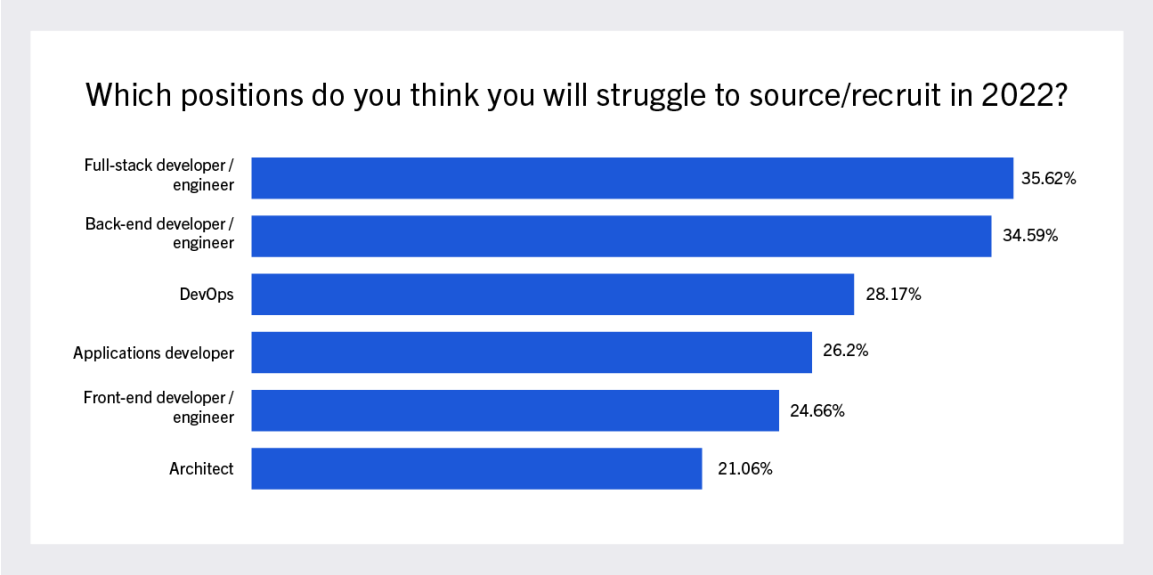
Now more than ever businesses run on software. That means that your team — whether you're a "software company" or not — likely includes software developers. Brand and product experiences are increasingly dependent on software, and delivering delightful, compelling experiences depends on having a talented, engaged technical team.

But if you've ever had to hire for a technical role, you know that finding the right person, with the skills needed to support your business and your applications — whether you're small or large — is easier said than done.

No matter how technical your business is, the road to building a great development team is full of potholes. Unfilled roles lead to a loss in productivity, overworked teams with low morale, and a lack of competitive edge. Even when you do hire a team of developers, it can be challenging to manage, train, incentivize, and nurture them to deliver to the best of their capabilities.

Want to hire a developer? So does everyone else.

Developers are critical to your business — so why not simply hire more? If it feels like finding good technical talent is tougher than ever, you're not imagining things. Software engineers are a hot commodity right now, and the market for hiring them is extremely competitive. A recent survey of over 18,000 developers and recruiting professionals, for example, revealed that recruiting for key engineering and DevOps roles in 2022 has been a major challenge.



That demand impacts everything from the cost of hiring developers and the timelines required to bring someone onto the team to the stakes for retaining your hard-won talent. IT and engineering staffing executives have reported that critical job roles are staying open for extended periods of time, and organizations must resort to more drastic measures to fill them. In 2020, average salaries for top engineering roles went up by 5% in the San Francisco Bay Area, 3% in New

York, 7% in Toronto, and 6% in London — continuing to grow despite the massive economic downturn and an overall drop in hiring demand due to the COVID-19 pandemic.

Developer? Engineer? Programmer?

You'll notice in this guide that we use the terms "developer" and "engineer" somewhat interchangeably. Does that mean there's no difference? Not quite.

Technical teams encompass a wide range of roles and expertise — far too many to list here. For the sake of brevity, in this guide we'll use "developer" to mean any member of your technical team. But these guidelines can be used as a reference for any role, from SRE to QA automation engineer to software developer and beyond!

To build a better technical team, create a developer-centric organization

To effectively attract and retain technical talent, organizations must think beyond compensation alone and make real changes to how they scale and maintain their engineering teams.

In this guide, we'll explore what makes developer-centric organizations successful. We'll also provide actionable advice on how to find, hire, and onboard technical talent to give them a head start at success — starting with fostering a team culture that keeps developers engaged.

PART 1

Building a Developer-Centric Culture

Great technical teams are built, retained, and scaled with strong developer-centric cultures. Ensuring that people feel safe, respected, and empowered to do their best work is essential to building any team, and developers are no exception!

Culture can be a tricky thing, though. It's not something that can be measured, and while it can't truly be enforced from the top down, a healthy culture requires the right environment to grow and scale. Before you start adding headcount, lay the foundation of a developer-centric team culture that devs will love to be part of.

What do great developer cultures have in common?

While many teams struggle to build great developer cultures, some organizations do seem to have things figured out. Let's take a look at some of the characteristics that developer-centric teams have in common.

They empower developers to work autonomously together

Enabling your team to work autonomously helps them feel ownership over their work. It also helps keep their work interesting and engaging over time. Give your team the agency they need to move quickly, collaborate effectively, and adopt new tools freely.

Keep in mind that "autonomy" doesn't mean that every developer is a lone wolf, acting on

their own. Rather, it means that the team has the flexibility to build and collaborate however they need to to meet the project's needs.

Spotify: Organizational structure designed for autonomy

As the engineering organization at Spotify grew, its early Agile processes and practices began to slow down development. Spotify shifted to a model of "Autonomous Squads," each of which has its own mission and end-to-end responsibility for whatever they're building.

The team has control over how they organize themselves and achieve their goals. This model helps developers at Spotify stay motivated, keeps decision-making moving fast, and minimizes handoff and waiting time. Learn more about Spotify's engineering culture »

Put it into practice

- ☑ Prioritize building a functional codebase
- ☑ Reduce technical debt
- ☑ Normalize documentation and code commenting
- ☑ Create clear handoff and well-documented processes
- ☑ Reduce friction with your developers' preferred tool stack

Diversity and inclusion are more than just metrics

Diversity is a buzzy word right now, and it can sometimes feel like organizations talk the diversity talk, but don't walk the walk. But hiring a more diverse set of candidates is only the first step toward building a diverse and inclusive organization.

Developer-centric organizations must give explicit, loud support to underrepresented folks within the company and user base. Developers aren't always eager or skilled at voicing their opinions. If the quietest person in the room feels adequately understood and represented, the rest of the team is much more likely to feel the same way.

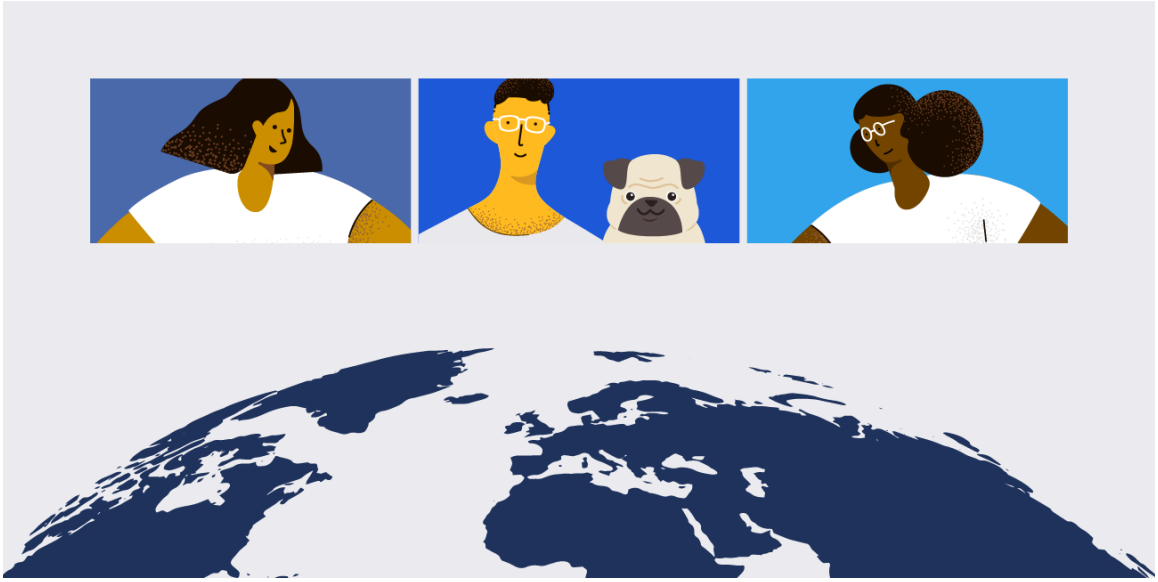
Asana: Programmatic diversity and inclusion initiatives that scale

From the early days, Asana committed to cultivating a diverse team. But they didn't simply set a hiring quota and call it "done." Instead, they built a fully-fledged, action-oriented program that starts with their hiring pipeline and processes, and continues throughout the entire lifecycle of employees at the organization.

Asana also hired a Head of Diversity and Inclusion to focus full-time on maintaining and extending their values as they grow the team. Now, the role includes partnerships with organizations that work with underrepresented demographics, benefits programs designed to bridge the gap for groups that sometimes get lost in the shuffle (e.g., mental health and parental leave), and employee support groups to help team members connect. Learn more about Asana's diversity and inclusion program »

Put it into practice

- ☑ Create spaces for team members to connect with each other
- ☑ Hire team leads that champion all team members
- ☑ Document your processes and measure your impact



They encourage career growth and provide a roadmap to success

Development work can be a grind. Far too often, developers get so bogged down in their day-to-day work that true career growth is overlooked. Great developer cultures counteract this in a few important ways:

1. **They provide clear pathways to growth.** They document what's needed to get to the "next level" and make that documentation available to every team member.
2. **They create space for growth.** They work to ensure the team has the right tools and processes in place to work efficiently and prioritize projects. They also hire appropriately to ensure the team isn't overloaded.
3. **They give every member of the team an advocate.** Their leaders speak up for folks who might not feel they have a voice and regularly check in with the team members to ensure they're moving forward and upward.
4. **They support non-traditional pathways.** They recognize that not everyone has the same background or career goals and create support systems for hiring and promoting people who don't fit the mold, but who have great skills and experience to bring to the table.

Patreon: Honoring a commitment to professional growth

Like many organizations in their early days, the Patreon team lacked clear criteria for growth within their engineering organizations. After realizing this, Patreon developed a comprehensive guide to leveling up engineers to provide a clear, actionable roadmap for team members looking to grow. This document helps support early-stage developers navigate the organization, and provides guidance for what they need to do to climb the ladder internally. Learn more about Patreon's engineer leveling »

Put it into practice

- ☑ Create opportunities for professional development — engagement with open source tools, training sessions for new technologies, funding for continued education, etc.

- ☑ Support regular knowledge-sharing activities, such as Lunch and Learns, pair coding, technical blogging, and process documentation, etc.
- ☑ Increase visibility into organization charts and promotional paths for all team members
- ☑ Encourage 1-on-1s with managers that include career growth and goals, not just day-to-day responsibilities

They leave nothing unwritten

Tribal knowledge can be very helpful — as long as you're a fully entrenched member of the tribe. For fast-growing teams — and especially for distributed ones — those things that everyone "just knows" don't always filter through the ranks as thoroughly as they might in a shared office. Even when that knowledge does make its way through the team, it doesn't always translate across roles, levels, and departments consistently.

Communication is essential to any team. For technical organizations, it's even more critical. Great documentation of everything from daily workflows and processes to career pathways and review cycles is essential to high-performing technical teams.

Keep in mind that this documentation doesn't appear out of thin air — everyone on your team should be empowered and encouraged to create, contribute to, and iterate on documentation of processes and best practices as part of their day-to-day work. Incorporating a new tool into your workflow? Add it to the docs. Found a way to streamline a repeatable process? Update the playbook. After a while, documentation will become second nature, and the whole team will benefit from it.

Great teams don't leave the team's work "unwritten" either. Public recognition for hard work, contributions, and growth can increase visibility into what everyone is doing and bolster morale before, during, and after working on the toughest problems.

GitLab: Increasing information accessibility and transparency for all

GitLab has built a wildly successful DevOps platform in just a few years — all with a fully remote, globally distributed team. At the core of their organizational operations is a documentation-first culture. Their single source of truth is an encyclopedic public handbook that would clock in at over 8,000 pages if printed in its entirety.

The goal of this approach is simple: to provide every member of the team with all the information they need to get their work done, no matter what time zone they're working from. Learn more about GitLab's handbook-centric documentation »

Put it into practice

- ☑ Consolidate all your how-tos, FAQs, and guides in one place
- ☑ Make sure your messaging platform is searchable
- ☑ Default to public channels for conversations — not direct messages
- ☑ Create playbooks or runbooks for repeated processes
- ☑ Have retrospectives as a regular part of your processes and workflows
- ☑ Include formal recognition into the rhythm of launches, work cycles, etc.

They use cutting-edge and open source technologies

A great developer can make the most of whatever technology you throw at them, right? Guess again. A StackOverflow survey of over 100,000 developers found that when it comes to deciding where to work, technology is a major deciding factor for developers, trailing only compensation and benefits as the top priority.

So, how do you know what tools are the right ones for your team? Part of the equation is letting them tell you! According to Google's DevOps Research and Assessment (DORA) program, DevOps teams that are empowered to choose their own tools perform better than those that don't have any say:

> *"Allowing teams to choose tools doesn't mean each team is given free rein to select any tool they want. Introducing technologies without any constraints can increase technical debt and fragility. However, when you combine tool choice with other capabilities—for example, a full view of the system, fast feedback, and the understanding that they are responsible for the code that they write—it helps your technologists make wise decisions about tools they will use and need to support."*

In fact, DORA's research has found that one of the most common indicators of job satisfaction is whether people have the tools they need to be successful.

Open source technology in particular is an essential part of many developers' preferred tech stack because of its flexibility and customizability. This preference points to an important aspect of empowering developer teams: They want and need tools that can adapt to their specific workflows and help them work as effectively as possible.

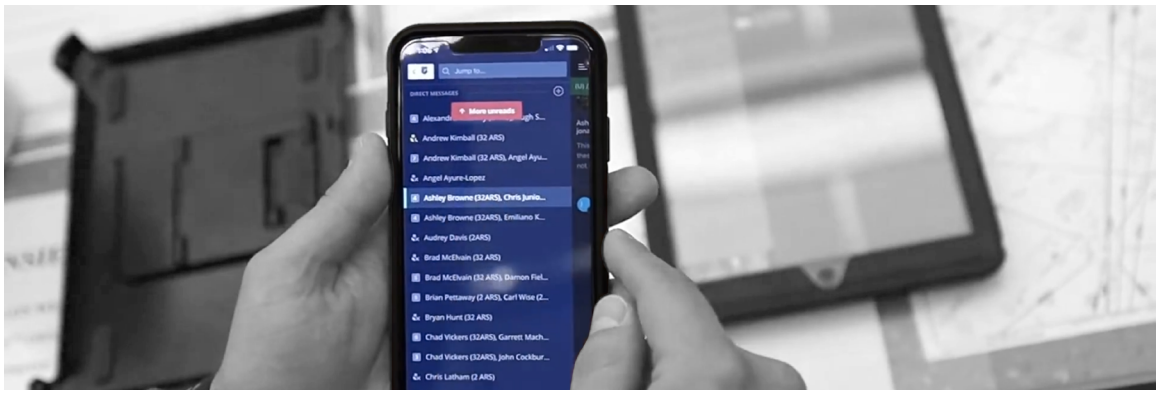
Did You Know?

The highest in-demand skill for software engineers in 2021 was Redux.js, landing engineers nearly 3x more interview requests than average. Experience with Google Cloud, AWS, React.js, Go, and Scala were also highly sought-after — all of these skills garnered at least 2x more interview requests for developers.

United States Department of Defense: Enabling open source tool adoption

The Department of Defense (DOD) might not be the first organization you think of when you think of high-performing tech organizations. But in the past few years, the DOD has gone through a serious digital transformation that has changed the face of software engineering for the entire organization.

Platform One, the U.S. Department of Defense's DevSecOps Enterprise Services team, adopted a new approach to streamline the approval and procurement of software solutions. Now, teams within the DOD are able to spin up new tools much more rapidly and cost-effectively than before, allowing them to leverage cutting-edge, cloud-based and open source tools that were previously unavailable to them. As a result, they've cut their software spending by millions of dollars and enabled their team to move many times faster than before by revolutionizing their approach to software development. Learn more about the Department of Defense's digital transformation »



Put it into practice

- ☑ Work with your team to understand the technologies they need and want
- ☑ Establish clear, quick procurement guidelines and processes
- ☑ Enable your team to experiment with open source tools and other cutting-edge technologies

THE BOTTOM LINE

Great developer cultures are cultures of innovation

There's no one "right" way to build a great developer culture, and it's impossible to completely overhaul your culture overnight. But even small changes can add up to a big difference over time. Former StackOverflow COO Jeff Szczepanski shared his thoughts on hiring great developers:

> *"[W]e do our best to give our programmers what they need, whether that's a high-end keyboard, three monitors, or the ability to work remotely. We also have an open-door policy for new ideas and feedback across the organization. We've found that this combination of tools and transparency makes all the difference.* ***The bottom line is that if you can't treat developers like critical stakeholders in your company's strategy, they'll find another company that will."***

At the end of the day, great developer cultures are cultures that encourage innovation. They give teams opportunities to think outside the box, be creative with their work, collaborate in unique ways, and give back to the community.

Adopt robust, flexible workflows, and tools that reduce toil. Eliminate timesinks and give the team the bandwidth to innovate.

Make "safe" spaces for people to bring up new, unusual, or "weird" ideas. Encourage projects like Open Source Fridays and hackathons that give developers space to think creatively.

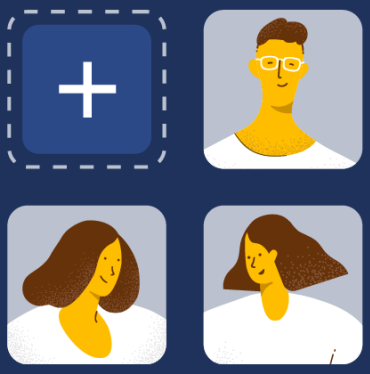
Prioritize work-life balance and provide opportunities for growth. Keep the team engaged, refreshed, and thinking about the future.

Now that you have a better picture of the type of culture you want to foster at your organization, it's time to start bringing in new people to help you realize your vision and get your technical team to the next level.

PART 2

Hiring Technical Talent:

Finding the Right People for Your Developer Team



You're working toward creating a culture that's compelling for a technical audience. But how exactly do you find and hire the right prospects? Developing a pipeline of high-quality candidates that turn into new hires and, eventually, productive members of the team requires a process focused on building personal relationships and trust.

Turning up the volume on your technical team pipeline

In a highly competitive job market, simply posting on general job boards (or on your own Careers page) doesn't cut it. You'll either end up with a flood of candidates to sift through — each of whom may or may not be a great fit — or a lack of applicants as your listing competes with thousands of others like it.

Want to shake up your candidate pipeline? It's time to make some changes to attract the best candidates.

Increase transparency to create interest

You've worked hard to establish a developer-centric culture, invest in high-impact methodologies, and leverage interesting technologies. Don't keep it a secret! Building a public reputation as a great place to work will help bring great candidates to you rather than forcing you

to seek them out. Encourage your devs to talk about their work and their teams, whether that means writing technical articles and blog posts, giving talks at industry events, or sharing their experiences with the community.

Hone your job descriptions and hiring team

According to seasoned HR executives, one of the biggest turnoffs for developers is feeling like HR and recruiting teams don't respect them or understand their skillset. Your recruiting team might not be technical themselves ("Is that a programming language or a Pokemon?"). Even so, you need to make sure that there's clear alignment on the requirements for the role and that your recruiters understand how to pitch the team to developers — from what your tech stack includes to what your developer culture is like.

Incentivize employee referrals

Who better to help you find your next great hire than the folks you've already brought onto the team? The developers who are already part of your organization are an excellent re-

source for finding new technical team members. After all, they know better than anyone what it takes to be successful within the organization, and they have a vested interest in hiring great colleagues. An employee referral program can provide the nudge they might need to proactively source candidates on your organization's behalf.

Connect with people who are in your community

From the end users of your product to the developers using the same technologies you do and beyond, your community might be an underutilized resource for finding the next member of your developer team. These folks are more likely to know your product, your tech stack, and even share cultural values with your team, making them "warmer" candidates than the average applicant. If your organization is involved in the open source community, you're in luck: Your contributors can be a great source of talent that's already invested in the success of your product.

Keep in mind that community engagement is an ongoing process — not something you can turn on and off when you need to hire. Get involved in (or host) industry events and conferences, create spaces for your community to interact with your brand and each other online, and keep the conversation going.

Expand your candidate pool

Do your minimum requirements include formal technical education? With the proliferation of bootcamps and other technical training programs, there's a huge pool of candidates with technical chops coming from non-traditional education backgrounds. In 2021, 43 percent of hiring managers stated that it is likely that they will fill a IT support or help desk position with someone who doesn't have a four-year-degree.

Teams that have gone remote also have a hiring advantage, as they can tap into a global pool of candidates rather than sticking to local job markets that might be oversaturated and highly competitive.

What makes a good technical hire?

There's no one checklist for the perfect technical hire. Your tech stack, culture, existing team, and development needs are all factors you need to consider when evaluating candidates. But the soft skills that are sometimes overlooked during technical candidate vetting can be highly impactful in the long-term success and happiness of your technical team.

Communication and collaboration skills. Does their collaboration style fit with the way your team works together? If your team is remote, are they great at written communication and asynchronous work?

Value alignment. Do their values and professional ethics map to the values of your organization?

Versatility and adaptability. Are they able to learn new technologies and processes as your needs grow and change?

Motivation and ambition. Are they excited to do their best work, learn, and grow alongside the rest of the team?

Creating an interview process that captures interest and retains momentum

You've identified some amazing candidates who are curious about joining your organization. That's a great start. But how do you nurture that interest and curiosity into excitement and commitment? Simple: You treat the folks you're hiring like you value their time and expertise — because you certainly should if you want to hire them!

Every organization's hiring process is a little different, but here are a few tips on creating an interview process that transforms candidates into new hires.

Key traits of a technical interview process

☑ **Keep everything moving.** Be realistic with your expectations. While it would be great to have every candidate meet every person on the team and spend plenty of time vetting them, candidates don't have time to commit hours and weeks of time to your hiring process. Aim for 3–5 hours of interviewing and make sure those hours are well-spent.

☑ **Keep conversations engaging.** Whiteboard demos and tests can feel tedious and performative for developers. Instead, focus on conversations and projects that help the candidate show off their skills and experience and give them a better understanding of the team, the role, and the organization.

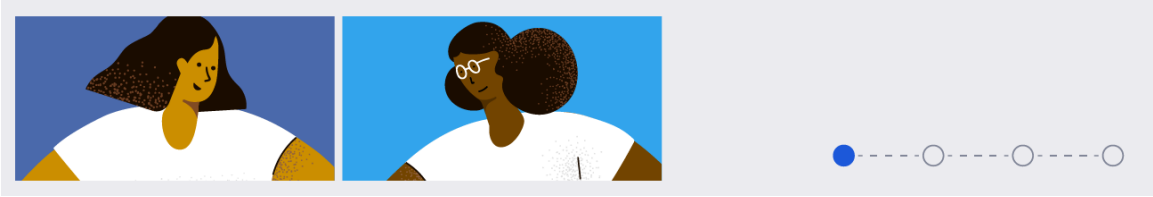
☑ **Communicate the process.** No one likes to be ghosted. Be clear about timelines and expectations at every stage of the process — especially if something is slowing things down.

☑ **Consider compensation.** Pay interviewees for their time especially if you're asking for work from them as part of the process.

An interviewing case study: What does a great interview look like?

So, what does a great technical interview process look like in practice? Here's a brief case study of what a developer's interview process might look like to help put those key traits in context.

The recruiter screen



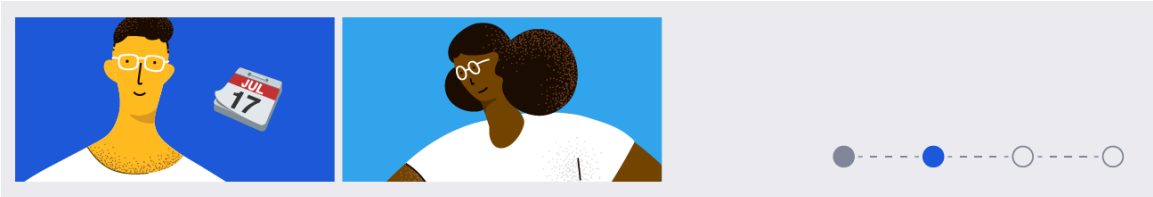
It's a dark and stormy…Tuesday morning, and Sarah is interviewing for a front-end develop-

er role. She hops on a Zoom call with her recruiter, who is the main point of contact for the interview process.

The recruiter has already discussed the role with the hiring manager and understands exactly what the team is looking for. She can answer Sarah's initial questions about the role and the team, and jots down a few notes to pass along to the hiring manager for the questions she doesn't know the answer to. The recruiter also lets Sarah know what to expect from the interviewing experience — from who she'll be speaking with to a general timeline for the process.

After syncing with the hiring manager, the recruiter promptly sets up the next round of interviews with Sarah.

Hiring manager, technical, and team interviews



Sarah meets with the hiring manager and a few other members of the team. She and the recruiter have worked together to carve out times that work for both parties and try to keep the interviews close together so this stage doesn't take too long.

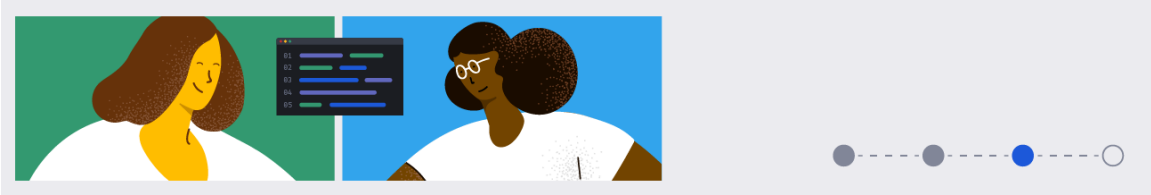
The hiring manager and the other members of the team that Sarah is meeting with have had time to review both the job description and Sarah's resume before their conversations, and each team member is tasked with evaluating specific areas of expertise. She speaks with her hiring manager, two developers, and a product manager — all of whom she would work with closely if she joins the team.

Oops! One of the developers on the interview panel had to cancel due to a conflict. The recruiter steps in to let Sarah know, and they reschedule so Sarah isn't left hanging out on a Zoom call by herself.

The hiring manager makes sure that Sarah's questions are answered, any concerns she has

about the role or the team are addressed, and that they understand what she is looking for in her next job. They make themselves available for follow-up conversations if Sarah requests them.

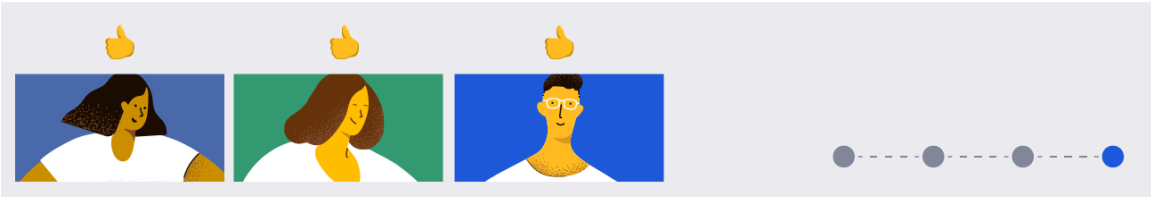
Pair programming interview



The team loved their conversations with Sarah, and she's ready to move onto the next stage. The recruiter sets her up with another developer on the team for a pair programming interview, where she and the developer work together to solve a problem. The interview gives Sarah the opportunity to show off her skills in action and get a sense of what it would be like to collaborate with a developer from the team.

Pair programming is a great alternative to a take-home project or presentation because it's closer to the actual daily workflow for Sarah's role. At the same time, it also helps keep the time commitment for the interview from ballooning out of control.

Making the final decision



Sarah's interviews are done! The team syncs up to discuss her performance, raise any concerns they have, and make a final decision. No matter what their ultimate choice is, the hiring manager or recruiter keeps Sarah looped in on what's going on. One of the interviewers needs an extra day to fill out their feedback? No problem. But the recruiter makes sure to reach out to Sarah and let her know that they haven't forgotten about her.

THE BOTTOM LINE

Attracting technical talent is about building connections

Navigating the process of finding and closing new technical talent can be challenging. But a thoughtful process can make all the difference — for both your existing team and any candidates they engage with.

While the ultimate goal of any hiring process is to fill an open position, the best recruiting and interviewing workflows are focused on checking boxes. They're designed to create a dialogue with potential team members that gives both sides a chance to make informed decisions about their next big move.

Build a network. Forge relationships with the people who might someday be candidates before you're actively looking for them.

Aim for transparency. Document and communicate processes to keep prospects and candidates informed from day one through the final offer.

Respect everyone's time. Keep interview processes as streamlined as possible and focus on creating positive experiences.

PART 3

Onboarding Technical Talent:

Setting up New Hires for Long-term Success

Everyone's been there at some point or another: You're new on the job, sitting in a meeting with a bunch of people you *sort of* know, and they're talking about a project you're *kind of* familiar with. But as they dig into the latest project's details, they might as well be speaking Ancient Egyptian.

While the interview process can feel like the biggest hurdle to bringing a new member of the team into the fold, getting a new hire up to speed often takes months, and — if mishandled — can leave your newest team member feeling like a perpetual outsider. This is even more important for remote team members who don't have the advantage of being able to connect with people in person in low-friction, unplanned ways.

What does a successful technical onboarding look like?

On average, it takes about six months for new hires to become productive members of the team. But that doesn't mean you can't beat the curve.

Equipping the newest member of your team with a game plan to help them onboard effectively will help ensure that their first few days, weeks, and months are successful. They might still feel like they're reading hieroglyphics, but at least they'll have the tools needed to decode them.

Rolling Up Your Sleeves

Why it's important

Developers want to build things. The first few days and weeks can be a lot of training sessions, paperwork, and getting-up-to-speed work that doesn't feel productive.

Giving them the opportunity to get their hands on the code and start contributing is one of the best ways to make them feel welcome out of the gate.

Putting it into practice

- ☑ Set small goals that ramp up
- ☑ Give them time and permission to start working on things
- ☑ Get them involved in processes and workflows early
- ☑ Provide documentation so they can self-start

Connecting with the Team

Why it's important

Your candidate met a handful of people during their interviews. Now it's time to really make them feel like part of the group — not just a guest.

Create space for them to get to know their new team both 1-on-1s and in larger team settings, especially in less structured forums than existing meetings. If you don't already have regular team lunches, happy hours, or game nights, now might be a great time to start a new tradition!

> *"Communities of practice play a huge role in assimilating new engineers.Connecting new hires to relevant communities of practice helps to quickly create a sense of belonging and share knowledge."*
>
> *— Gartner, Inc., "How to Create a Frictionless Onboarding Experience for Software Engineers," Manjunath Bhat & Thomas Murphy, 17 February 2021*

Putting it into practice

- ☑ Set up 1-on-1s with peers, managers, and mentors
- ☑ Have hiring “buddies” for the first few weeks
- ☑ Hold dedicated “welcome” meetings or sections during existing meetings
- ☑ Create regular events for the team to socialize

🔍 Knowing Where to Find “Stuff”

Why it’s important

Visibility into processes, workflows, and historical information makes all the difference between feeling like the “new kid” and an old pro. After all, most folks don’t want to spend their time endlessly searching for information or guessing on the right way to get things done.

Make sure that new members of the team have everything they need to work productively starting on day one.

Putting it into practice

- ☑ Provide a clear onboarding roadmap with links to key information
- ☑ Prioritize keeping processes documented and easy to find
- ☑ Encourage communication in public channels rather than DMs

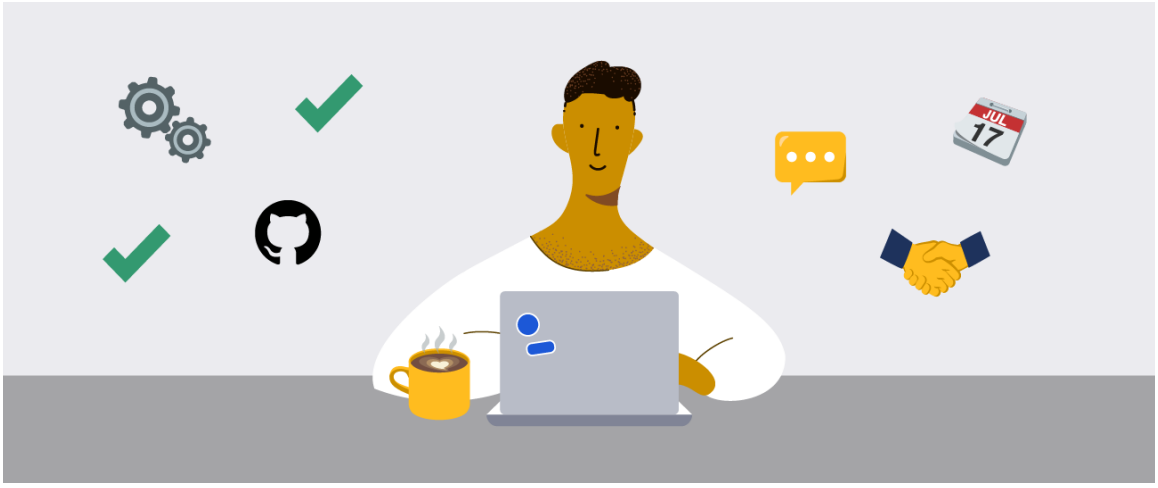
Technical Onboarding Playbook

No matter what role your new hire is onboarding for, there are a lot of moving parts involved in bringing a new person onto your team. Luckily, a clearly documented, well-defined on-boarding process can make that transition a little less overwhelming for everyone.

Every organization's onboarding processes will be a little different, but there's a common pathway for success for most technical teams. Check out our Technical Onboarding Playbook — based on the one we use here at Mattermost to onboard new members of the Engineering team — to learn what the first four months can look like for a new developer.

Week 1: Focus on environment setup and introductions

- ☐ Set up equipment and environment
  - ☐ *Get your hardware and set up dev environments, clone the repo, and set up accounts. Expense any equipment needed and finish HR setup.*
- ☐ Meet the team
  - ☐ *Meet with a different team member for 30 minutes each day*
- ☐ Meet with your dev lead
  - ☐ *Meet for at least 10–15 minutes each day*
- ☐ Attend weekly team/company meetings to get the feeling for them
  - ☐ *Sprint planning, company all-hands, dev meetings, dev hangouts, etc.*
- ☐ Join important messaging channels and project boards
- ☐ Read and understand the org's core values
- ☐ Work on 1–3 small tickets to get used to dev processes
- ☐ Introduction at the R&D meeting

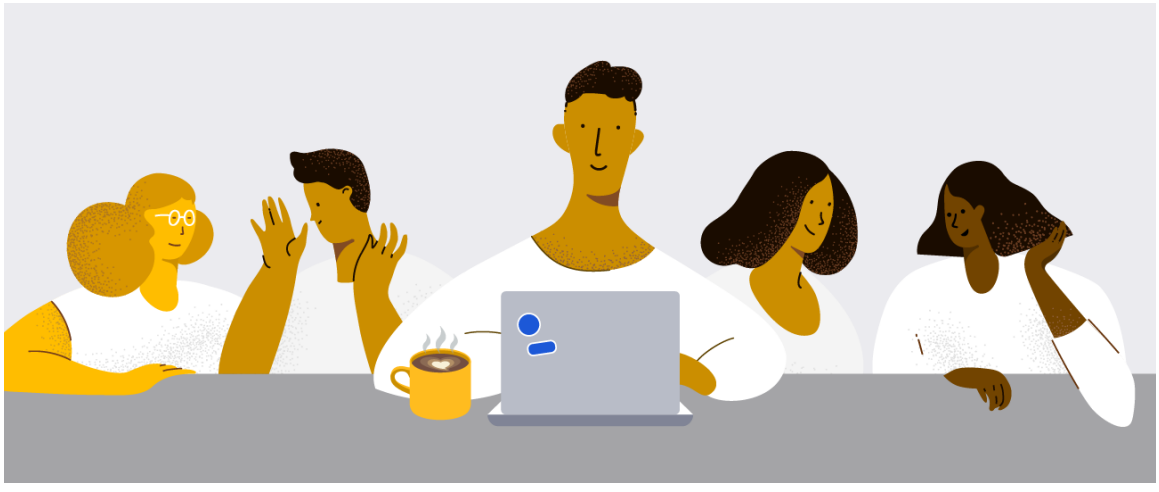


Week 2: Focus on digesting information dump

- ☐ Continue getting used to processes and workflows
  - ☐ *Attend some spec/design/technical meetings; your dev lead should send you invites to these meetings*
  - ☐ *Do some code reviews; your dev lead should assign you to some*
- ☐ Introduction meetings with a different team member each day
- ☐ Mentor for the week is a senior teammate
- ☐ Meet with dev lead 2–3 times in the week and review company values
- ☐ Work on 1–3 small tickets
- ☐ Introduction at the company all-hands

Weeks 3–4: Focus on solidifying role in the team

- ☐ Start taking a more active role as a member of your dev team
- ☐ Begin to participate in some spec/design/technical meetings
  - ☐ *Provide feedback/ask questions, have an active role in the meetings*
- ☐ Mentor for each week is a different teammate
- ☐ Start regular 1-on-1s with dev lead
- ☐ Work on some medium-sized tickets



Weeks 5–8 (Month 2): Work on your first project as dev owner

- ☐ Begin work on your first relatively large task and be the dev owner of it
- ☐ Act as the dev owner in the spec/design/technical meetings
- ☐ Write up a brief technical specification if necessary
- ☐ Work on small/medium tickets each sprint alongside the larger task

Weeks 9–11 (Month 3): Work on more and/or larger projects as dev owner

- ☐ Take what you learned being dev owner on your first project and improve upon it
- ☐ Act as dev owner and write technical specs for more and larger projects
- ☐ Work on small/medium tickets to fill in gaps
- ☐ Complete deliverables in a more timely fashion
- ☐ Attend a “Tech Moonshots” developer meeting

Week 12: Informal performance evaluation

- ☐ Dev lead will give you your performance evaluation
- ☐ A self-survey will be sent to you
- ☐ Surveys will be sent out to your peers and you are:
    - ☐ *Evaluated on what’s being done well, what could be improved*
    - ☐ *Evaluated on core values*

Weeks 13–16 (Month 4): Act on your performance evaluation and focus on broader engagement

- ☐ Begin acting on any feedback given during the performance evaluation
- ☐ Start taking a more active role in the community and customer engagement
    - ☐ *Proactively answer questions in channels*
    - ☐ *Create Help Wanted tickets*

THE BOTTOM LINE

Lay the groundwork for success

Your newest team member wants to start making meaningful contributions right away, and it's your job to give them everything they need to hit the ground running. The right tools and onboarding processes are essential to getting people up to speed and working quickly, which helps set the stage for a happier, more productive team in the long run.

Create a roadmap to success. Your new hire has a lot of catching up to do. Make it easy for them to navigate your organization with clear documentation, checklists, and tutorials.

Prioritize relationship-building (especially for remote teams!). The first few months can be an information overload, and finding a connection with team members can mean the difference between sinking and swimming. Plug your new hire into the existing team both as a group and one-on-one.

Establish check-ins early on. Your new hire might be eager to dig in. But that doesn't mean you should let them tackle every challenge on their own. Check in with them on a regular basis to make sure they're on track.

Final Thoughts

Building a Thriving Technical Team

Whether you're adding your first few software engineers or are expanding and diversifying your technical operations, building your technical team is an ongoing journey. By laying the foundations of a stellar developer-centric culture, investing in the tools and processes that enable highly-productive collaboration, and building a recruiting process that captures the interest of the best candidates, you'll be able to grow a dynamic, innovative technical team that brings your business to the next level.

We hope this guide has provided you with actionable steps for building your technical team, and that you're ready to venture out and find your next great hire! Want to learn more about enabling your technical team to be happier and more productive? Check out our more resources on developer productivity best practices and tools on the Mattermost blog. While you're at it, sign up for our newsletter and receive content like this directly in your inbox.

About Mattermost

Hundreds of thousands of developers around the globe trust Mattermost to increase their productivity by bringing together team communication, task and project management, and workflow orchestration into a unified platform for agile software development. Founded in 2016, Mattermost's open source platform powers over 800,000 workspaces worldwide with the support of over 4,000 contributors from across the developer community. The company serves over 800 customers, including Samsung, Nasdaq, SAP, European Parliament, and the United States Air Force, and is backed by world-class investors, including Redpoint, YC Continuity, Battery Ventures, and S28 Capital. To learn more, visit www.mattermost.com.

Mattermost

Mattermost.com

🎉 9