TECH RECRUITMENT 101

Candidate Experience in Tech Hiring

PART OF THE *TECH RECRUITMENT 101* SERIES

Candidate Experience in Tech Hiring

How to make every hiring stage feel clear, respectful, and worth the candidate's time

CONTENTS

Introduction

I wrote this because candidate experience gets treated as something soft when it is usually a process problem. When a candidate feels confused, repeated, or ignored, the cause is often simple: the role was not defined clearly, the interview loop was not designed properly, or nobody owned the next update. After building recruiting teams a few times, I have seen how fast trust breaks when the process looks improvised from the outside.

That matters even more in tech hiring, where candidates are asked to invest real time and attention. They read job posts closely. They notice when recruiter messaging and hiring manager messaging do not match. They notice when assessments feel generic, when interviewers repeat each other, and when nobody can explain whether AI tools are allowed. The problem is not only that these moments feel frustrating. It is that they make candidates question the quality of every decision that follows.

So I wanted to make this practical. This guide looks at candidate experience stage by stage: where trust breaks in tech hiring, how to run a candidate experience audit, what communication reduces anxiety, how scheduling and interview design show respect, how to handle decision timing and feedback loops, and how to review the process so people would willingly go through it again. These

are the patterns I see most often in real searches, and they are the ones recruiters can fix without adding layers of unnecessary process.

This mini-book is part of the Tech Recruitment 101 series, which I put together for recruiters who already know the basics and want clearer operating habits. My goal here is simple: help you audit your process step by step, spot the points where trust usually drops, and fix them in a way that makes hiring feel clearer, faster, and more human for the candidate and easier to run for the team.

Why candidate experience breaks so easily in tech hiring

Candidate experience is not a brand slogan. It is the sum of small operational choices. In tech hiring, candidates judge us less by what we say and more by whether the process makes sense.

Engineers notice broken process fast. If the role sounds different in the recruiter call and the hiring manager interview, trust drops. If two interviewers ask the same questions, trust drops again. If nobody can explain the next step or timeline, the candidate assumes the team is making it up as it goes.

That is why I treat candidate experience as a discipline. We are not managing vibes. We are managing clarity, timing, and handoffs. Good candidate experience means the process feels

coherent from the outside because it is coherent on the inside.

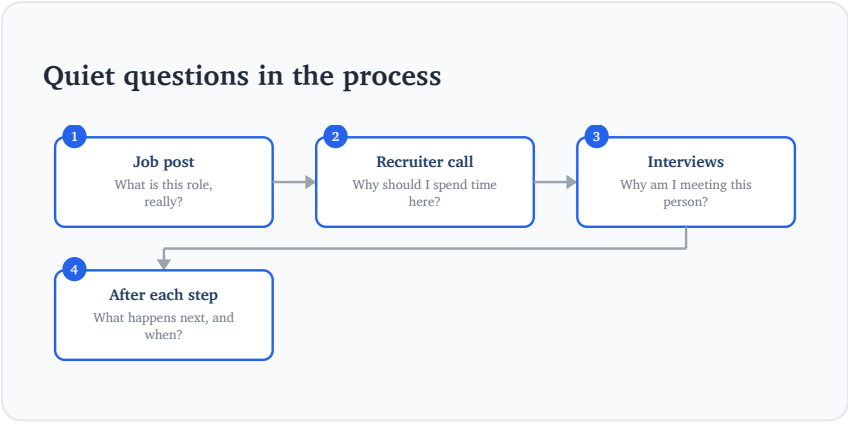
This matters even when the candidate does not get the job. A rejection can still feel fair if the process was clear, relevant, and respectful. People can accept “no” more easily than confusion.

The research supports this bluntly. In Checkr’s 2025 Hiring Disconnect report (https://checkr.com/resources/articles/hiring-disconnect-2025-report), 63% of job seekers said employers were not transparent about what to expect in interviews, including rounds, projects, and evaluation criteria. In the same report, 83% said ghost hiring had created an extreme lack of trust in employers’ hiring practices. That is a process control problem.

Most bad candidate experience starts inside the company. The hiring manager wants one profile, the panel wants another, and the recruiter gets told to “see what’s out there.” The interview loop, meaning the full set of interviews and assessments a candidate goes through, grows because nobody agreed on what must be tested. Feedback arrives late because nobody owns the decision. Internal confusion leaks outward with perfect efficiency.

I find it useful to think of candidate experience as a chain of promises. Some promises are spoken. “You’ll hear from us by Friday.” “This interview will focus on system design.” Others are silent. “This role is real.” “This panel knows why you are here.” “Your interviewers have read your background.”

At each stage, ask: what question is the candidate quietly trying to answer right now?



If a stage does not answer its question, trust weakens. A job post should define the work, not list every technology the team has touched since 2017. iHire's 2025 State of Online Recruiting (https://www.ihireveterinary.com/resourcecenter/employer/pages/the-state-of-online-recruiting-2025) found that 60.5% of candidates want hiring timeline transparency in job postings, 57.1% want salary ranges, and 36.2% want must-haves clearly separated from nice-to-haves.

Technical hiring adds extra ways to lose trust because the process has more moving parts. A backend engineer may meet a recruiter, a hiring manager, a peer, and an engineering leader. If those people are not aligned, the candidate feels every gap.

Structure helps. HireVue's 2025 guide to structured interviewing (https://www.hirevue.com/wp-content/uploads/2025/07/07_23_StructuredInterviewGuide-1.pdf) summarizes research showing structured

interviews predict success better than unstructured ones. Clear scope is not bureaucracy. It is mercy.

AI raises the bar further. Candidates now assume some outreach, scheduling, and screening may be AI-assisted. That is reasonable. LinkedIn's 2025 Future of Recruiting (https://business.linkedin.com/hire/resources/future-of-recruiting) describes generative AI as a way to automate time-consuming work so recruiters can spend more time on candidate experience and advising hiring managers. LinkedIn's 2026 talent research (https://news.linkedin.com/en-us/2026/LinkedIn-Research-Talent-2026) found that 93% of recruiters plan to increase AI use, while 66% plan to increase AI use for pre-screening interviews. If part of the process feels automated, the human parts must feel sharper, not thinner.

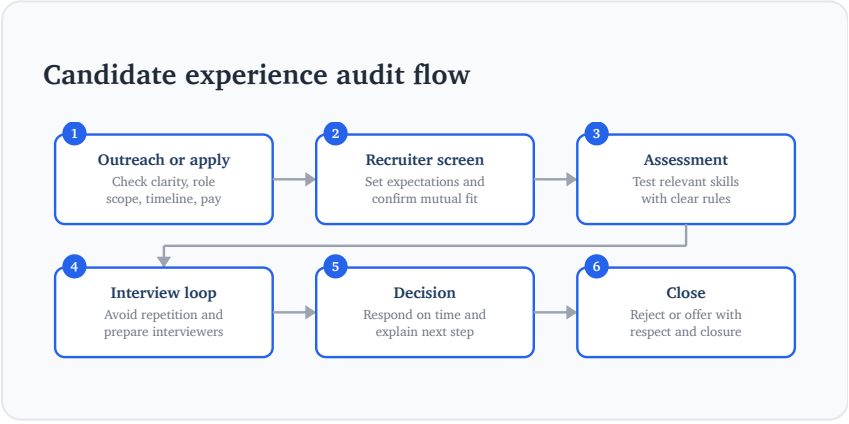
That means simple things matter more. Explain what an interview is for. State whether AI tools are allowed in an assessment. Tell candidates who they will meet and why. Confirm decision timing before they have to chase you for it.

Keep this working rule for the rest of the book: candidate experience breaks when the process stops answering the candidate's next reasonable question.

The candidate experience audit by hiring stage

Most candidate experience problems start before the interview, when nobody checks what the process will feel like from the candidate's side.

That is why I like a stage-by-stage audit. Map each hiring step and ask four questions at every stage: What does the candidate need to know? What effort are we asking for? Who owns the next step? When will the candidate hear from us?



Use the audit like a pre-flight check with the hiring manager. If nobody can explain why a step exists, delete it or fix it. If two interviewers ask the same questions, redesign the loop. If the team wants a take-home exercise but has no review standard, stop there.

Start with outreach or application. The candidate should know what the role is, what it is not, what the likely process looks like, and what happens after they apply. What usually goes wrong is vague language, stale roles, missing salary information, and silence. Checkr found that 66% of job seekers had applied to roles that looked open but later turned out to be inactive. Checkr's Hiring Disconnect 2025 report (https://checkr.com/resources/articles/hiring-disconnect-2025-report)

Then look at the recruiter conversation. A good screen checks fit and removes uncertainty. By the end of that call, the candidate should know the hiring steps, who they will meet, what will be assessed, and when to expect an update. Checkr found that 63% of job seekers think employers are not transparent about what to expect during interviews. Checkr's Hiring Disconnect 2025 report (https://checkr.com/resources/articles/hiring-disconnect-2025-report)

Next comes the technical assessment. This stage breaks trust fast because it asks for real effort. The task should match the job, the time ask should be reasonable, and the rules should be explicit. If you are hiring Site Reliability Engineers (SREs), test how they would handle incidents, meaning service problems or outages in live customer-facing systems, and how they make system trade-offs, meaning choices between things like speed, reliability, cost, and complexity. For example, you might ask: "A core service is timing out for customers after a new release. What would you check first, how would you reduce impact, and what would you change to prevent it happening again?" Do not send a generic

puzzle set because it was handy. HackerRank's 2025 guidance says developers prefer practical coding challenges over abstract puzzles, and recommends clear AI-use rules in more job-like environments. HackerRank's 2025 interview guidance (https://www.hackerrank.com/writing/build-better-real-world-interview-questions-2025)

The interview loop is where consistency matters most. Each interviewer should have a clear area to assess. The candidate should not have to repeat the same story four times. HireVue's 2025 guide summarizes research showing structured interviews predict success better than unstructured ones. HireVue's Guide to Structured Interviewing (https://www.hirevue.com/wp-content/uploads/2025/07/07_23_StructuredInterviewGuide-1.pdf)

Decision and close need their own checks because this is where many teams fail quietly. iHire's 2025 ghosting survey found that 53% of candidates said they had been ghosted by an employer. iHire's 2025 ghosting survey (https://www.ihireengineering.com/resourcecenter/employer/pages/53-percent-of-job-seekers-have-been-ghosted-by-a-potential-employer) The real issue is ownership. Who sends the update? By when? What do we say if the hiring manager is delayed?

I use a simple test. These are not a second framework. They are the signs that your four audit questions are being answered well. At each stage, ask whether the candidate gets four things: clarity, preparation, consistency, and closure. If one is missing, trust drops.

Audit checks by stage

- ✓ Does the candidate know the steps and timeline?
- ✓ Is the effort asked proportional to the role?
- ✓ Do interviewers know what they assess?
- ✓ Are AI-use rules stated where relevant?
- ✓ Does one person own every next-step update?
- ✓ Will every candidate get a close, not silence?

If you do this audit well, you will spot the real problems fast: duplicate interviews, slow decisions, unclear assessments, missing ownership. None of that requires a grand transformation.

Communication that reduces anxiety instead of creating it

Good candidate communication does not mean more communication. It means fewer gaps, fewer guesses, and fewer surprises.

When people do not know what is happening, they fill in the blanks themselves. In iHire's 2025 survey, 53% of candidates said they had been ghosted by an employer, most often right after applying or after an early conversation. In Checkr's 2025 research, 63% said employers were not transparent about what to expect in interviews, and 62% said lack of feedback after applying hurt their confidence and mental health (iHire (https://www.ihireengineering.com/resourcecenter/employer/pages/53-percent-of-job-seekers-have-been-ghosted-by-a-potential-employer), Checkr (https://checkr.com/resources/articles/hiring-disconnect-2025-report)).

The fix is simple. At every stage, answer four questions before the candidate has to ask. What is this step? Who will I meet? How long should it take? When will I hear back?

Your first contact sets the tone. Most bad outreach is vague or sounds mass-produced. A short message works if it proves you understand the job and gives the candidate a reason to care. "I'm hiring for a backend engineer role focused on API performance and service reliability" is useful. "I have an exciting opportunity at a fast-growing company" is wallpaper.

Role framing matters just as much. If the role is truly backend-heavy, say that. If it needs a product-minded engineer who works on both front-end and back-end parts of the product, say that instead. Do not smooth over tradeoffs to get a call booked. If the recruiter describes one job and the hiring manager describes

another, trust drops immediately. After building recruiting teams a few times, I can tell you this is one of the fastest ways to make a decent process feel sloppy.

You prevent that problem before the search starts. Ask the hiring manager what this person will spend most of their time doing early on, what skills are necessary on day one, and what interview evidence will count as strong. Then turn that into one plain-language version of the role that everyone uses.

Interview prep is where respect becomes visible. Candidates should not have to reverse-engineer your process from calendar invites. Tell them the sequence, purpose, and format of each step. If they are meeting a Site Reliability Engineer, or SRE, explain why that person is in the loop. If one interview tests debugging and another tests system design, say so. If the coding exercise allows AI tools, say that. If it does not, say that too.

Weak vs strong candidate messages

MOMENT	WEAK	STRONG
First outreach	Want to chat about a role?	Backend role. Focus on APIs and scaling. 30-min intro?
Interview prep	You'll meet the team.	You'll meet the hiring manager and two engineers. One interview covers architecture, one covers debugging.
Delay update	Still waiting on feedback.	The team needs two more days because one interviewer is out. I'll update you by Thursday.
Rejection	We went another direction.	We chose someone with deeper production incident experience. Thanks for the time you invested.

Status updates should be boring in the best way. If you promised an update by Friday, send one by Friday. If there is no decision yet, say that plainly. A short note is enough.

Delay messages need the same specificity. "Things are taking longer than expected" says almost nothing. Better: "The panel finished, but the hiring manager is traveling Monday. We will debrief Tuesday and I will write to you that afternoon."

Rejection communication is where many teams become evasive. You do not need to write an essay. You do need to close the loop clearly and on time. If you can share a brief, honest reason, keep it tied to the role.

AI is changing the mechanics, not the standard. LinkedIn's *Future of Recruiting 2025* says generative AI helps recruiters automate repetitive work so they can spend more time on relationship-building and candidate experience (LinkedIn (https://business.linkedin.com/talent-solutions/resources/future-of-recruiting/)). Let AI give you a draft structure for outreach, prep notes, or follow-ups. Do not let it flatten the details.

Use templates, but make them specific where it counts. Name the stack. Name the interviewers. Name the next date. Name the reason for the delay. Every candidate message should remove a question, not create a new one.

Scheduling, prep, and interview design: where respect becomes visible

Candidates notice respect when the process works: the calendar invite is correct, the interviewer is ready, and the loop ends before it turns into a small endurance sport.

Scheduling is the first operational proof of competence. Offer real windows, not a single slot dressed up as flexibility. If the candidate is in another time zone, send options in their local time and double-check daylight saving changes.

Good scheduling also means matching the process to the decision. Do not build a five-step loop for a role that could be assessed in three focused conversations. If two interviewers are both testing communication, or three people are all asking basic technical questions, the process is too long and too fuzzy.

Prep is where fairness becomes practical. A candidate should know the format, expected length, tools, and focus areas before the interview. If you are scheduling a coding screen, meaning a live technical interview where the candidate solves or discusses coding problems in real time, say whether they will write code live, work in a shared document, use an online editor, or discuss an existing code sample. That is different from a take-home exercise, which they complete on their own time, or a discussion-based interview, which focuses on how they think and explain

rather than writing code during the call. If you are hiring a Site Reliability Engineer (SRE), say whether the conversation will test incident response, system trade-offs, or troubleshooting.

Candidate prep note should cover

- ✓ Interview format and length
- ✓ Names and roles of interviewers
- ✓ Platform and tools required
- ✓ Skills being assessed
- ✓ Whether AI tools are allowed
- ✓ What happens after the interview

Interviewers need prep too. Each interviewer should know what they are evaluating, what evidence they need, and what other interviewers already cover. Without that structure, candidates get the same question three times from three different people.

This is one reason structured interviews matter. HireVue's 2025 guide summarizes longstanding research showing that structured interviews predict job performance better than unstructured ones, with validity estimates of about 0.51 versus 0.38 (HireVue (https://

www.hirevue.com/wp-content/uploads/2025/07/07_23_StructuredInterviewGuide-1.pdf). In plain language, those numbers refer to how well an interview method predicts later job performance. Higher is better, and 0.51 versus 0.38 is a meaningful gap, not a trivial one. Better structure is kinder to candidates and better hiring practice.

A simple interview plan works well. One interviewer tests technical depth. One tests collaboration and communication. One hiring manager tests role fit and scope. If you need a practical exercise, make sure it answers a question conversation alone cannot answer.

Assessments are useful when they are relevant, proportionate, and clearly explained. HackerRank's 2025 guidance says developers prefer practical challenges over abstract puzzles, and recommends job-like assessments with clear AI-use rules (HackerRank (https://www.hackerrank.com/writing/build-better-real-world-interview-questions-2025)).

AI has made this more urgent. Candidates will reasonably ask what is allowed in a take-home, coding task, or live exercise. If your answer is vague, you create uneven conditions. Greenhouse's 2025 hiring report found that 27% of candidates say they have never seen an employer policy on AI use in hiring (Greenhouse (https://www.greenhouse.com/blog/greenhouse-2025-workforce-hiring-report)). State the rule before the interview, not after you see the result you do not like.

A systems design interview without warning does not test only systems thinking. It also tests whether the candidate guessed your process correctly. That is not rigor. It is sloppiness with a confident tone.

If you want a better candidate experience, audit the invisible mechanics. How many steps does this role really need? What does each interviewer own? What does the candidate know in advance? Where could time zone mistakes, duplicate questions, or unclear AI rules create avoidable stress? From my years doing this, candidate experience usually improves when the process becomes shorter, clearer, and more deliberate.

Decision timing, feedback loops, and the final stretch

The end of a hiring process is where trust gets tested. Candidates can forgive a hard no. They struggle with a vague maybe that drags on while nobody seems to know what happens next.

If you want to fix it, start before the interviews begin. Set the decision path early. Who gives input? Who makes the call? When does the debrief happen? When will the candidate hear from us? If those answers are fuzzy, the process will drift when emotions are highest.

Most delays come from loose ownership. One interviewer submits notes late. A hiring manager wants one more chat. The debrief invite vanishes into the calendar void. None of that feels internal to the candidate. It feels like indifference.

I treat the final stage like an operating rhythm, not a special event. Book the debrief before the interviews happen. Name one decision-maker. Set a response window for the candidate. Then tell the candidate that window in plain language. Checkr's 2025 Hiring Disconnect report (https://checkr.com/resources/articles/hiring-disconnect-2025-report) found that 63% of job seekers feel employers are not transparent about what to expect in interviews.

A good debrief is short and evidence-based. Each interviewer should answer the same questions: What did you test? What evidence did you hear or see? What risk remains? Would you hire this person for this role?

When the team cannot decide, do not hide behind silence. Tell the candidate what is happening. A useful update has three parts: why there is a delay, what the next checkpoint is, and the exact date of that checkpoint.

> ✓ **Simple rule for delays**
>
> If you cannot decide yet, explain why, say the next checkpoint, and give a date.

Speed matters because silence lands harder than most teams think. In iHire's October 2025 ghosting survey, 53% of candidates said they had been ghosted by an employer at some point, including after interviews. iHire's survey (https://www.ihireengineering.com/resourcecenter/employer/pages/53-percent-of-job-seekers-have-been-ghosted-by-a-potential-employer)

When a candidate has a competing offer, compress the process honestly. Ask for the candidate's deadline, check what is truly possible, and come back with a real plan. If you need one more conversation to decide, say so. If the team cannot move in time, say that too.

Internal disagreement needs the same discipline. Bring the discussion back to the scorecard, meaning the interview feedback form or rubric that lists the skills being assessed and the evidence or ratings from each interviewer, and the role requirements. Which concern is real? Which evidence supports it? Which gap could the candidate learn quickly on the job?

Feedback is the part people overcomplicate. In many companies, detailed feedback is limited by policy or legal caution. Fine. You still owe the candidate a timely close. Checkr's 2025 report (https://checkr.com/resources/articles/hiring-disconnect-2025-report) found that 62% of job seekers said lack of feedback after applying hurt their confidence and mental health.

A rejection should be clear, kind, and final. Thank them for the time they invested. Tell them the decision. If allowed, give one brief, job-related reason.

AI can help a little here. Automated reminders can chase feedback forms, flag overdue debriefs, and nudge people before deadlines slip. LinkedIn's Future of Recruiting 2025 (https://business.linkedin.com/hire/resources/future-of-recruiting) frames the value well: automation saves time so recruiters can spend more of it on relationship-building and candidate experience. But software cannot own a hard conversation, and it cannot repair a broken promise after the fact.

In the final stretch, candidates are not asking for perfection. They are asking for signs that someone is steering the process. Give them dates. Give them decisions.

Building a process people would willingly go through twice

Do a candidate experience review after every few searches, or sooner if one process goes sideways in public. Keep it light. You do not need a task force or a six-month transformation plan. You need honest notes and a willingness to remove friction.

Start with the points where trust usually breaks. Look at where candidates drop out, where offers die, where the same questions keep coming back, and where recruiter notes start to sound familiar. If three candidates ask whether there is a take-home exercise, your prep is unclear. If people vanish after panel interviews, the process may be too long or too vague. If offer declines mention “slow” or “confusing,” believe them.

The fixes that matter are usually boring. Cut one interview that repeats the same signal. Send better prep before technical rounds. Define a service level for updates, even if the update is “we are still deciding.” Write clearer briefs for interviewers so candidates do not answer the same question four times. Faster debriefs help too.

What to fix first

- ✓ One repeated interview
- ✓ Prep that answers common questions
- ✓ Update timing by stage
- ✓ Clear interviewer briefs
- ✓ Debrief within 24 hours

Use data, but do not worship it. Survale's 2025 benchmark research (https://survale.com/10-key-takeaways-from-the-2025-candidate-experience-benchmark-research/) is useful because it treats candidate experience as an operating discipline, not a brand slogan. Checkr's 2025 report (https://checkr.com/resources/articles/hiring-disconnect-2025-report) found that 63% of job seekers felt employers were not clear about what to expect in interviews. That gap rarely needs a grand solution. It usually needs clearer stage descriptions, better closeout discipline, and fewer surprises.

AI can help with the boring parts in the right way. LinkedIn's 2025 Future of Recruiting report (https://business.linkedin.com/hire/resources/future-of-recruiting) makes the point well: automation should give recruiters more time for candidate relationships, not less. Use

AI to summarize survey comments, group repeated complaints, or spot patterns across recruiter notes. Do not let it make the judgment call for you.

Candidate experience is what the process feels like from the outside when the inside works properly. Clean handoffs, clear expectations, and quick decisions are not glamorous. They do make people think, “I’d go through that again.”

Conclusion

If I had to reduce this whole guide to a few operating rules, they would be these: define the role in plain language, make each hiring stage answer the candidate’s next reasonable question, keep assessments relevant, prepare both candidates and interviewers properly, and never let decision timing drift without an update. Most candidate experience problems do not come from bad intentions. They come from vague ownership, duplicate interviews, weak prep, and silence at the wrong moment.

So the next step is not to redesign everything at once. Run a simple audit on one live role. Check the job post, the recruiter screen, the assessment, the interview loop, and the close. Look for one broken promise at each stage. Then fix the process issue behind it. In most teams, a shorter loop, clearer prep, explicit AI-use rules, and firmer update ownership will improve candidate experience faster than any new policy deck.

This book sits alongside the other guides in the Tech Recruitment 101 series for a reason. Candidate experience does not stand alone. It depends on role clarity, interview design, hiring manager alignment, and good decision habits. If this topic exposed friction in those areas, that is a useful signal. Follow it. The best hiring processes do not feel impressive from the inside. They feel clear and fair from the outside.

Glossary

- **AI (Artificial Intelligence)** — Software that can automate or assist with tasks like outreach, scheduling, screening, and drafting candidate communications.
- **AI-assisted** — A process or task that uses AI tools to support human work rather than being done fully manually.
- **API (Application Programming Interface)** — A way for different software systems to communicate with each other; in hiring context, recruiters may see this used to describe backend engineering work.
- **APIs (Application Programming Interfaces)** — The connections that let software systems exchange data or functionality with each other.
- **Architecture** — The high-level structure and technical design of a software system.
- **Assessment** — A hiring step used to evaluate a candidate's skills through tasks, exercises, or tests.
- **Automated reminders** — Software-triggered notifications that prompt people to complete actions like submitting interview feedback or attending debriefs.
- **Automation** — The use of software to handle repetitive work with limited manual effort.
- **Backend engineer** — An engineer who works on server-side systems, data flows, performance, and reliability behind the product interface.

- **Back-end** — The server-side part of a product or system that handles logic, databases, and processing behind the scenes.
- **Calendar invite** — A scheduled meeting invitation sent through calendar software, often used to organize interviews.
- **Candidate experience audit** — A stage-by-stage review of the hiring process to find where candidates may experience confusion, friction, or loss of trust.
- **Candidate prep note** — A message sent before an interview that explains format, timing, tools, interviewers, and what will be assessed.
- **Coding exercise** — A technical task used to evaluate how a candidate writes, discusses, or approaches code.
- **Coding screen** — A live technical interview where a candidate solves or talks through coding problems in real time.
- **Competing offer** — A job offer from another employer that may affect a candidate's hiring timeline.
- **Debugging** — The process of finding, diagnosing, and fixing problems in software or systems.
- **Debrief** — A discussion after interviews where the hiring team reviews evidence, compares feedback, and decides what to do next.
- **Decision-maker** — The person who has final responsibility for making the hiring decision.
- **Discussion-based interview** — An interview focused on how a candidate thinks, explains, and reasons, rather than having

them write code live.

- **Engineering leader** — A senior technical leader responsible for engineering teams, direction, or management.
- **Engineers** — Technical professionals who build, maintain, and improve software systems or infrastructure.
- **Feedback form** — A structured document interviewers use to record what they assessed and how the candidate performed.
- **Front-end** — The user-facing part of a product that people directly interact with.
- **Generative AI** — AI that can create new content such as text, summaries, drafts, or messages based on prompts.
- **Ghost hiring** — A hiring pattern where employers appear to be recruiting but fail to communicate clearly or close the loop with candidates.
- **Ghosting** — Ending communication with a candidate without explanation or closure.
- **Hiring manager** — The manager responsible for the open role and usually a key decision-maker in the hiring process.
- **Incident response** — The actions taken to investigate, manage, and resolve a live technical problem or outage.
- **Incidents** — Service problems or outages that affect live systems or users.
- **Interview loop** — The full set of interviews and assessments a candidate goes through for a role.
- **Interviewer briefs** — Written guidance that tells interviewers what to assess, what evidence to look for, and how their

interview fits into the process.

- **Live customer-facing systems** — Production systems that real customers actively use and that directly affect their experience.
- **Live exercise** — A real-time interview task completed during the interview rather than independently beforehand.
- **Live role** — An actively open position that a company is currently hiring for.
- **Must-haves** — Skills or qualifications that are essential for doing the job successfully.
- **Online editor** — A web-based coding tool used during interviews or assessments.
- **Outages** — Periods when a system or service is unavailable or not working properly.
- **Panel** — A group of interviewers involved in assessing a candidate.
- **Panel interviews** — Interview stages where multiple interviewers evaluate a candidate, either together or as part of a coordinated loop.
- **Peer** — A teammate at a similar level who interviews a candidate to assess collaboration or technical fit.
- **Plain-language** — Communication written in simple, clear wording without unnecessary jargon.
- **Platform** — The software or system used to run interview steps, coding tasks, or virtual meetings.

- **Pre-screening interviews** — Early-stage interviews used to quickly assess whether a candidate appears to meet basic role requirements.
- **Process control problem** — A failure in ownership, structure, timing, or communication within a workflow.
- **Product-minded engineer** — An engineer who thinks not only about technical execution but also about user needs, business goals, and product impact.
- **Production incident experience** — Practical experience handling real problems in live systems used by customers.
- **Production systems** — Live systems that are running in real use and serving customers or internal users.
- **Profile** — The type of candidate background, skills, and experience a team is trying to hire.
- **Recruiter screen** — An early conversation where the recruiter checks mutual fit and explains the hiring process.
- **Release** — A new version of software that has been deployed or made available.
- **Rubric** — A structured set of criteria used to evaluate a candidate consistently.
- **Scorecard** — An interview evaluation form that lists the skills being assessed and captures interviewer ratings or evidence.
- **Screening** — The early review of candidates to decide whether they should move forward in the process.
- **Search** — A hiring effort for a specific role or candidate profile.

- **Service** — A software component or system that performs a specific function within a larger product or platform.
- **Service level** — A defined standard for how quickly a team should respond or provide updates.
- **Service problems** — Technical issues that reduce the quality, speed, or availability of a system.
- **Shared document** — A collaborative file that multiple people can view or edit during an interview or assessment.
- **Site Reliability Engineer** — An engineer focused on keeping systems reliable, available, and operational in production.
- **Site Reliability Engineers (SREs)** — Engineers who focus on reliability, incident handling, troubleshooting, and balancing system trade-offs in live environments.
- **SRE** — Short for Site Reliability Engineer.
- **Stack** — The set of technologies, tools, and frameworks used to build and run a product.
- **Structured interviews** — Interviews that use a defined format, consistent questions, and clear evaluation criteria across candidates.
- **System design** — The practice of planning how a software system should be structured to meet requirements like scale, reliability, and maintainability.
- **Systems design interview** — An interview that tests how a candidate would design a software system and think through technical trade-offs.

- **Systems thinking** — The ability to understand how parts of a technical system interact and affect each other.
- **Take-home exercise** — A technical task a candidate completes on their own time outside the live interview.
- **Task force** — A temporary group assembled to work on a specific project or problem.
- **Tech hiring** — Hiring for technical roles such as engineers and other technology-focused positions.
- **Technical assessment** — A hiring step designed to evaluate technical skills through a job-related exercise or problem.
- **Technical depth** — The level of expertise and understanding a candidate has in a technical area.
- **Technical questions** — Interview questions that assess job-relevant technical knowledge or problem-solving.
- **Technical rounds** — Interview stages focused mainly on assessing technical skills.
- **Technical term** — A word or phrase with a specific meaning in a technical or engineering context.
- **Tech Recruitment 101** — The name of the book series referenced in the text for recruiters learning to hire in technical environments.
- **Time zone** — The local time region a candidate or interviewer is in, which affects scheduling.
- **Timeline transparency** — Clear communication about the expected sequence and timing of hiring steps.

- **Timing out** — Failing to respond within the expected time, often causing a request or service to fail.
- **Trade-offs** — Choices between competing priorities such as speed, reliability, cost, and complexity.
- **Troubleshooting** — Investigating and resolving technical issues in a system or service.
- **Unstructured** — Lacking a consistent format or defined evaluation approach, often referring to interviews.
- **Validity estimates** — Statistical measures showing how well an assessment method predicts future job performance.

About This Book

About the Author

Natalia Romanova

Natalia is the founder of Talantrix, an applicant tracking system for tech recruiters. She began her career as a software engineer, later moved into engineering management, and eventually led several software companies, including ones she founded. Along the way she repeatedly had to build technical recruiting capability, and over time ended up learning a lot about hiring engineers—more than she ever expected to.

About Talantrix

Talantrix is an intelligent applicant tracking system built by recruiters, for recruiters. Designed specifically for technical hiring, it helps recruitment teams automate the busywork — from CV parsing and skill-based search to pipeline management — so they can focus on what matters most: connecting with top engineers and making great hires.

Learn more at https://talantrix.com (https://talantrix.com)

The Tech Recruitment 101 Series

A practical book series for recruiters who work with engineering teams. Each volume tackles a specific challenge in technical hiring — from screening candidates to understanding code — with clear explanations, real-world examples, and actionable frameworks.

Revision 5.1 — 6 April 2026

Tech Recruitment 101

A practical book series for recruiters who work with engineering teams. Each volume tackles a specific challenge in technical hiring — from screening candidates to understanding code — with clear explanations, real-world examples, and actionable frameworks.