Senior Engineer Jobsearch

Land your dream offer in 100 days

Abstract

If you're an engineer aiming for an offer at a large company, how you do the job search can make a six figure difference. This course will teach you all we learned on our job search landing senior offers.

Contents

Introductions

Course Introduction

Joe: Hey everyone, my name is Joe.

Stepan: And my name is Stepan.

Joe: Thank you for investing in our course! We're excited to help you land your dream offer. In this section, we'll tell you a bit about us and what you can expect from this course.

Stepan:

- Software engineer with 7+ years experience
- Recently signed an L6 offer
- I started my career by building a company — interesting part it was in China.
- After that I worked on Wit.ai, initially as the 2nd hire as a startup, and then at Facebook through the acquisition.

Joe:

- Software Engineer with 5+ years experience
- Recently signed L5 offer
- Previously worked at a couple startups and Facebook

Stepan: The knowledge from this course comes from our 100-day adventure, going on a job search as senior engineers. We went on a combined 20+ onsites.

Joe: We found out that, though there's a lot of information about job searching, some of the most key concepts were missing. Things like, communicating your narrative, or communicating your level through system design

Stepan: We started sharing this with our friends, and began to notice real changes in their performance — we saw 30-100% compensation differences.

Joe: Our goal in this course is to give you the roadmap and the lessons you'll need to complete a phenomenal job search.

Stepan: To use this course effectively, start with the study guide. There, you'll learn how you can layer the content, to get the best out of the course.

Joe: Throughout most sections you'll see homework. The homework is what's going to truly make the difference.

Stepan: If you follow along, and truly give it an honest effort, you'll come across more confident to recruiters, significantly improve your performance in the interview, and grow as an engineer.

Joe: We're here for you — if you have any feedback, please let us know.

Stepan: We put our heart into this, and we hope you love this course. Without further ado, check out the study guide, and start your journey!

Audience

This course is designed for the following candidates:

- Software Engineers specializing in front-end, back-end, or full-stack web development
- Located in major tech hubs in the US
- Looking for individual contributor roles, ranging from junior to staff level (E4 - E6 at Google/Facebook)
- Interviewing at pre-IPO or public tech companies (Uber, Lyft, Airbnb, Facebook, Google, Amazon, etc.)

Study Plan

To best utilize this course, we've prepared a 3-month study plan for you

W1: Foundations, Key Concepts, Ramp-Up

- Watch and complete the homework for Foundations, Key Concepts, and Ramp-Up

At this point, you should have a strong narrative, a team, and a list of companies. You should have reached out to your referrals, and began to set up recruiter screens

W2-3: Recruiter Screens, Start Preparing for Technical Screens

- Watch and complete the homework for **Recruiter Screens**
- Watch and complete the homework for the relevant technical screens interviews. (Algo, and UI if you are a UI engineer)
- Start working every day on Algorithm Interview and/or UI Interview prep.

At this point, you should be talking to recruiters and scheduling technical screens for W7. You should have a todo list, where you knock off technical-screen prep every day.

W4-6: Technical-Screen Prep, System-Design Prep

- Watch System Design Prep, and start the process

At this point, you should have scheduled all technical screens. You should be working on algorithm and system design prep every day, and have had at least 2, ideally 4+ practice interviews.

W7-8: Technical screens, system design prep

- Complete your technical screens
- Keep studying system design
- Watch **Onsites** video

Once you complete your first technical screen, evaluate where you are at. If you are struggling after a couple screens, move the other screens 3-4 weeks further, and re-start with W4-6.

W9-10: Onsites

- Keep focused, keep fed, and eyes on the prize

At this point, you'll be very busy with onsites, so no homework. Keep a log of every onsite you do, and lessons learned. If you are struggling or are uncertain of anything refer to the right section, or reach out to us.

W11-14: The Offer Stage

- Watch the Offer Stage Module

You're in the final mile now. Time to finish strong. Complete the homework, and keep up the vigilance. Once you sign, send us your notes, and if you have any suggestions for the course!

Mental Game

Mental Game Introduction

Here we will talk about the mental game behind the job search. These are the beliefs that underly every single interaction you have. We'll go over the kind of attitude, narrative, outlook, and commitment you'll need to set your job search up for success. When you get this right, the tactics just take care of themselves — how you perceive others, how others perceive you, and the kinds of actions you take come naturally. When this is off, it's hard for any tactic to work.

Are you ready? Without further ado, let's master the mental game!

Attitude

Most people dread the job search. They may think they are being judged, or they may think that, the process is silly. You'll hear "Interview questions don't measure your real performance", or "Agh, I just want this to be over with".

This kind of attitude causes a self-fulfilling prophecy. When you dread something, or if you think it's silly, you spend very little time on it. If you spend very little time on it, your results will reflect — you'll see negative interview experiences, negative recruiter conversations, and it will all feed back into your attitude. Now, if this makes you take the first job you see, because "you want this to be over with", and it's not the best opportunity for you, you've just paid a high price for this attitude.

Now, how do the best candidates see this?

The best candidates love the job search. They see the job search as a unique, infrequent opportunity, which let's them level up their career, level up their compensation, and level up their community.

Why?

Level up your career.

This is the point where you have the most leverage to affect what you do. Do you have any gripes in your old job? Anything you dream of doing? If you wanted to take on more responsibility or work on different problems, this is your chance!

Level up your Compensation

Compensation is accelerating. You might be very surprised with what the market rate is now. Your responsibilities could have grown too, which means you might be mis-leveled at your current job. This is your chance to see a step function in your salary — from our friend circles we've seen people get between 30% - 100% higher packages.

For Leveling up your Community

As you go through the job search, you will meet new people. All the managers you meet, all the interviewers, all the recruiters, they now know you more intimately, and they will have become part of your community.

Plus, as you go through the search you will be relying on your friends and colleagues, and those relationships will strengthen. For example, if a light connection refers you to their company, they've now become a much more deeper

connection. With all of that, you can see a path to get excited about the job search.

Now, there are still problems — sure algorithms may not directly align with job duties, but you can see that the whole process makes you better. You may dislike being judged, but you can see this as a way to truly understand what the market wants, and grow into that

For your homework, I want you to really think about how would you describe your previous approach to the job search? How are you seeing it differently now? Do you have any hangups still? What are you doing to work through them?

Change your attitude, and it will make everything else easier.

Homework

Question for you:

How would you describe your previous approach to the job search? How are you seeing it differently now? Do you have any hangups still? What are you doing to work through them?

Narrative

Now, that you *love* the job search, let's talk about how you see your career. If we were to ask you "why did you join your last job?" What would your answer be? How about "What did you do there", or "Why are you leaving?"

Most people spend *very little time* thinking about these questions. Let's go over some examples.

Why did you join your last job?

It may be a story of some accident – "My friend worked at this company, so I asked him for a referral, and now I'm here"

If you answered with an accident story, the recruiter may think that you had no real purpose to your choice, so you don't care deeply about your career.

Why are you looking for your next job?

You may have had a negative experience, and answer: "I disliked the environment", or "I wasn't growing anymore." Or perhaps you answer with something generic "I want to grow more", "I want a change, and I want to try something new"

If you answer with a negative story, it could make the recruiter wonder — was it your coworkers, or was it you? If you answer with a generic story, the recruiter will think perhaps you don't care about their career

How do the best candidates answer these questions?

The best candidates form strong answers, on where they came from, and where they are going in their career. They form a **narrative** that demonstrates **conviction** and **direction**. They demonstrate a purpose driving their choices, and how the work they did helped them grow. They show a north star vision which looks like an exciting journey.

The best candidates also demonstrate a framework for their choice — they are looking for specific kinds of growth, and they are considering *multiple* options to realize that growth — that can even involve staying at their current company.

When a recruiter hears all this, they begin to sense that **they have landed on a star.** Your criteria may be the perfect fit for them. This make the recruiter excited to close you, and even better, they can now imagine **a path to convince you to join.** They begin to show you how working with them fits the framework for your choice. Furthermore, they can pass this information to hiring managers

and get them excited too. All of a sudden you have a company that is thrilled to interview you.

By forming this narrative you also make it **easier for your community to help you.**

Imagine you meet up with an old mentor and are asking them for advice. If you're "looking for a change" — how can they help you? They can't just open their rolodex and send a hundred emails.

However, if you a paint a story, and show them your *ideal* next company, now it's easy for them to help — you have almost written the email they're going to send to introduce you!

You may be wondering though — what if my story isn't exciting?

This is often a mirage. If you spend a few hours to think about what your north star vision is, you'll begin to see a story unravel — and you'll get very excited about it.

If you don't believe us, let's *try it out.*

In your homework, we want you to spend 2 hours, and give us your answers to "Why you joined your last job", "What you did", and "Where you're going".

For demonstration, Stepan has included an example with his answer

As you answer these questions, you'll uncover the exciting path you've been on, and be ready to start the job search strong.

Homework

Take a moment to reflect about your journey. What are your answers to these questions?

- Why did you join your last job?
- What did you do there?
- Why are you leaving?
- What are you looking for in your next job?

Example Narrative

Below is an example of a narrative document we recommend you make early on in your job search process. This is the one Stepan wrote for himself.

Why did you join your last job?

I loved startups, and I was building a service business with my co-founder in China. We were beginning to be successful, but I realized two things. First, we wanted to solve ambitious and large problems, and both of us knew that this business is not one of those. Second, I realized I was growing in sales and marketing, but I was no longer growing technically. With that, we decided to wind down the business. I decided that I would dive deeper technically, and go where the biggest problems are being solved — San Francisco.

I found a company where all of us were weird in the same way. We were solving an ambitious machine learning problem, with technology that proved to be very well chosen. We were one of the first .ai domains, one of the first docker users, and one of the pioneers of Clojure — a functional language in Lisp.

A year in, we got a call from Facebook, they acquired us to apply our expertise in productizing NLP within messenger. I was worried at first, it was a big company, but I soon found that I *loved it* — we were given enormous support and latitude to solve problems that I will be proud of for the rest of my life.

What did you do at your last job?

During my time at the Startup, I was the primary owner of the UI in Wit. I migrated our app from Angular to React, and built the Tagger — a rich-text-editing piece that let users annotate data.

When I joined Facebook, for the first 2 years I was the UI lead on Oz – a service that facilitated "customer service on steroids". We built a system that a team of 60 agents used to answer questions. We helped agents automate as much as possible — from detecting parts of the conversation, to logging every single mouse movement, to phone calls to emails. This is the time that I reached my zenith in UI engineering, and gave talks around the world on React.

After that, I went on a 2-year long mission.

The first, was to find product-market fit internally for Wit. I patented, tech-led, and launched two products, which integrated Wit with different orgs across Facebook.

Second, I owned, what I termed "The Great Migration". This was a multi-phased migration, spanning over a year and a half and 6 people, with the goal of moving Wit business logic from our legacy infrastructure to Facebook.

Why are you leaving your job?

I finished The Great Migration in November. When I finished, I opened my

eyes to breathe again. I had finished the commitment I had made, and I knew that the next commitment I would make would have a similar commitment of a few years. I decided then, before diving deeper, I should reflect — Do I want to dive deeper at Facebook, do I start a company, or do I want to join another company?

What are you looking for in your next opportunity?

I know that at the end of the day I want to start a company. At first, I thought that I would start one in November. I realized though, that I didn't yet have a *burning* vision, for something that *had* to be done as a company. I also realized, that there could be a 2-year commitment I could make, where I could grow considerably at the next company. There were 3 dimensions:

Technical Growth: I have gotten to work on technical problems that I am very proud of, but there's still room on my technical mountain for growth. I could see myself getting excited about leading teams to build distributed systems and heavy client-side applications.

Product Growth: I was fortunate to have immense support at Facebook to build new products. However, one thing with building an API for NLP is that it's not often highly critical to the customer. I could grow building a product that is essential for customers, and essential for the business.

People Growth: This, I want to continue from my time at Facebook. I want to work with people who are on a mission, and know why they come into work each day. I want us to become a mafia, to support empower each other. Spending 2 more years growing my community, will pay immense dividends down the road.

Community

Now that you **love** the job search and you know *exactly* what you're looking for. Let's take a look at **how** to approach the job search.

Most people look at the job search as an independent activity. They think up the job that they want, they apply to them, they prepare the best they can, negotiate the best they can, and then choose the best they can.

Our question for you is this: How often do you see amazing work done by **only one** person?

Truly great work happens as a team. You are part of an interdependent network, so let's try to shift your view that way. Instead of thinking about your search alone, **let's build a team for your job search.**

Your team will consist of:

- Your referrals — People who will vouch for you. They will be your supporters throughout the stages.
- Your peers — People who are on the same road as you, and have either completed or on the job search right now. They can give you invaluable advice and information.
- Your mentors — Your previous managers, or people you look up to — they'll be instrumental in showing you what is possible, and what opportunity to ultimately choose.

As you go through the search, you'll often feel the desire to go at it alone — there may be a job you like but you only have a light connection with someone who works there.

We want to encourage you to *make yourself vulnerable, and reach out.* You'll be surprised with what happens.

For the homework, we'll *build your team.* Follow those steps, and you'll be well on your way to truly doing the job search interdependently – the results will follow!

Homework

1. Create a brain dump of potential mentors and peers
2. Reach out to 1-2 mentors, and set up a meeting
3. During those meetings, share your narrative with them. And ask them what they think about the opportunities you are considering. Ask:
    - Would they do anything differently?
    - What do they think about your plan of action?
    - Do they have ideas for companies that sound great? Note: Do not look for a referral here — if they offer, let them know you are still in exploration phase, and will update them in a bit

4. Reach out to 1-2 peers, who either completed a job search or are on their way. Ask them for a 1:1 to discuss:
    - What surprised them about their job search?
    - What companies and teams did they *really* like?
    - What tools are they using to prepare?
    - What do they think of your narrative?
    - Do they have any suggestions for companies that match it? Again, do not look for a referral here — if they offer, let them know you are still in exploration phase, and will update them in a bit

Commitment

You have now learned to *love* the job search. You have a *strong* narrative and are on an exciting journey with your career, supported by a team of peers and mentors. Now we reach the final step in the mental game — what kind of *commitment* will you make to the job search?

You can divide up the search into 3 parts:

- The Preparation — the studying, scheduling, and referrals
- The Execution — all the screens and onsites
- The Offer Stage — choosing and negotiating your offer

How much time should you dedicate to each, and at what level of focus should you approach them?

Preparation

You *could* do this part-time, however, it would take *significantly* longer to reach the *top* bar in algorithms and system design. To get a sense, we would expect about 4 weeks of full-time study to get there. If you do this part time, it may be harder to schedule practice interviews with your friends, and this is one of the best ways to learn.

Execution

You could do screens and onsites part time. However, you may end up reducing your ability to negotiate, reducing your chance to find the best opportunity. For example, part time you may only be able to do one or two onsites a week. This means that you will either:

- Only do ~4 onsites, reducing the number of opportunities you see, potentially losing the best one
- Stagger your onsites across multiple weeks which may mean your offers will land at different times and make it harder to negotiate and get the best compensation

Offer Stage

It's definitely possible to negotiate while doing other things. However, expect the negotiation phase to take multiple weeks, with multiple meetings. Why?

Once you've gotten the offer, it's time to choose, and it's your turn to interview the company. You may want to meet potential managers and teams multiple times. This is not only helpful for choosing the *right* opportunity, but it also helps during the negotiation. If your potential manager *really wants* you, the

recruiter now has further support for their case to the compensation team to help close you.

We've seen that focusing on your job search *full time* can make a significant difference in the opportunities you see, and the kinds of offers you close. We see two common objections to this

- It will cost a lot of money to not work for 3 months. This is true, however, we feel strongly that at least in the San Francisco market, the difference in offers will be much greater than this.
- Lack of confidence one will find another job. We think this is a false belief — if you are doing great at your job, you are in hot demand and there will be an option you're happy with.

Now, if there are reasons you need to do part time, you can still make it work. We suggest looking over the study plan, and customizing it to your schedule. However, we strongly recommend you aim to schedule all the onsites within a 2 week window — this will help you land offers around the same time, and it will make a world of a difference during negotiations.

With that, we've reached the end of commitment — you now have the attitude, the direction, the community, and the commitment to start off your job search strong. Onwards!

Homework

Question for you:

How will you approach the job search? When will you study? What will you do about the execution phase?

Communication

Communication Introduction

What do you think your goal is when talking to a recruiter? In our experience, most people spend very little time thinking about this. Some will say it's just a "performance" — they feel like they're going through the motions

From our video on attitude, you can get the idea of the self-fulfilling prophecy this will start — it's hard to get good at something that has no meaning.

However, the most common answer is this – "I try to be myself."

With the recruiter, they will talk to them as a friend, and let them lead the conversation where it goes.

Now, this is on the right track. Being yourself is good — actually, it's critical — if you pretend, it'll come through, and at the end of the day, you want to work a place where they want to work with *you.*

However, there's a there's a BIG misconception here —

The misconceptions it that *being you* is actually coming through to the interviewer.

What if we told you, that when you thought you were being yourself, you were not actually understood?

It would be like, if you flew to Beijing and tried to strike up a conversation. You could be speaking, but no one would understand a thing you're saying! — they don't speak your language.

There is a *language* in the interview process, and a *purpose* behind what recruiters and interviews are trying to do. You *can* be yourself, but if you don't know this language, you lose the ability to communicate. In some cases, you may be communicating the *opposite* of what you think you are.

In this module, we'll learn this language. This is the language of signal, scope, and leveling. Once you learn this, you'll see every single interaction in the job search differently — you'll know the purpose, and have the tools to communicate *exactly* what makes you a phenomenal candidate, in each step of the process.

Are we ready? Let's get into this.

Signal & Scope

The first step to learning the language of the interview is to understand the other side of the table – what are interviewers trying to do?

If you were dropped inside a hiring committee (the place where they decide your offer) you would get a clue, because they would use a few words, over and over and over again to describe your performance.

The main word? **signal**

The interviewer's job is to get as much **signal** as they possibly can.

Signal is information which demonstrates to the interviewer your specific skills and knowledge.

This lets them see what kind of experience you had, what kind of impact you can have, and the kind of leadership you can take on if you join.

The interesting thing — this signal is divided up — There are different kinds of signal, and interviews are designed to parse them separately.

For example:

- Coding signal. This answers — how deeply does this person understand, and how effective are they at actually coding? This is parsed through the algorithm interview — If you're asked "how do you merge sorted lists" — the kind of tradeoffs you mention, the solution you choose, the way communicate your solution, and the way you implement it — will reveal to the interviewer your coding signal.
- System signal. This answers — is this person experienced and capable in coming up with and leading a large technical system? This is parsed through the systems design interview. If you're asked to implement Google Photos — the way you drive the conversation, the tradeoffs you mention, and the solutions you propose will reveal your systems signal
- People signal. This answers — is this person experienced and capable in either working with, or leading, a group of people. This is parsed through the experience interview. Here you are asked: tell me about a conflict you had, the *kind* of conflict, the *size* of the conflict, and the *way* you resolved it. Your answers to these will reveal your people signal

Finally, there's signal that permeates throughout — and that is called **scope.**

Scope answers the following:

- What kinds of problems is the person going to be solving?
- Are they going to lead a team, a team of teams, or re-architect our entire company?

You want to demonstrate to the interviewer the scope that you are capable of owning. You do this through describing:

- The kinds of problems you solved — did you architect one system, or own a suite of systems?
- The kinds of conflicts you resolved — did it involve one team, or multiple orgs?

Based on your answers, you will be leveled a certain way (discussed in the next section).

Conclusion

You now know the language of the interview. The purpose for each interview is to extract signal, and the signal that shows our seniority the most is called scope.

When you're in an interview, **you need to keep these concepts top of mind.** It's like you're on a broken telephone — unless you focus on the conversation, the signal won't go through.

In the next sections, we'll learn exactly *how* to communicate signal and scope.

Homework

No real homework here — we want to get you thinking — if you've had an interview already, how do you think you've been communicating? Do you think you have been sending the right signals, or were you missing something?

Leveling

So far, we learned about signal and scope, and we learned that our goal in the interview is to *maximize signal* and to communicate as much as possible, exactly where we're at with engineering, technical leadership, and people leadership.

Now, once we've sent that signal — what do interviewers do with this information?

Once they've gotten signal, they'll aim to **level** you — this is the job title and responsibilities you'll have at the company.

Most large companies, have converged to a similar leveling system, and calibrating yourself to the right level, can be one of the most leveraged actions you take in the job search.

The level changes the game in two ways —

First, the right level defines what you do day to day — your level sets your expectations, and the kind of scope you take on — If you want to be solving large and ambiguous problems, it's much simpler to do that when you come in at a senior or staff level. By getting this right, you are starting off your role strong.

Second, level is the biggest deciding factor in compensation. Once your level is set, your compensation is determined by a range for your level, and where you fall in that range. By changing where you level is at, you can massively shift your compensation.

Homework

Spend about 1 hour looking through the leveling doc. Answer, which section most speaks to the kind of work you have done? Now take that level, as well as the one above it, and for as many points as you can, list briefly the work you've done that matches, or why you believe that you match that.

Our main advice for you — be ambitious, and let your work speak for itself — the more evidence you have, the more conviction you will have, and the higher the signal you will be able to convey. If you notice that you are close to the level above what you thought, you may want to think hard about going for it!

Leveling Guide

This document goes over each level you need to be aware of, as well as the technical and people expectations behind them.

We suggest using the document in two ways:

1. **Form a conviction for your level**

   As we discussed in the homework for **Communicating Signal**, you'll want to go over these levels, and get a sense of where you stand. You should have examples from your experience that match the description in these levels, and give you a strong sense of where you are at in your career.

2. **Get a sense of the compensation ranges**

   For each level, we've included the alternate names that most companies use. Go on levels.fyi, and take a peak at the compensation ranges for the companies you are aiming for. When time comes to communicate numbers to your recruiter, having accurate knowledge and concisely conveying it, signals strength and experience.

New-Grad

- **Alternate names: E3, L3, SDE 1, IC 1**
- **Experience Range: 1-2 years**

The New Grad level is given to people who join right out of college. This is often *the only* time where companies assign this level. **If this is not your first job search**, carry on to the level above.

Technical Expectations You complete *tasks defined by others* with speed, resourcefulness, and proactivity. The tasks you take on are often *guaranteed* to have a solution, and can be done within a week or two.

You can take ownership over these tasks and deliver them *on time* with *little guidance*. To do this, you use a combination of your technical ability: navigating the codebase, debugging, and your resourcefulness: researching, reaching out to people, asking for help with clear questions.

People Expectations The main people dimensions at this level is *how you handle feedback*, and *how proactive you are*.

You learn quickly and are constantly searching for feedback. When you get it, you take it positively, and look for tangible ways to improve.

When a task is confusing, or you reach an impasse you don't know how to solve, you demonstrate positivity and proactivity by asking for help, and remaining confident in your ability to complete the task.

You communicate well, and are proactively updating everyone around you about the timeline for your tasks when it changes.

Live Leveling Guides

- Patreon Leveling Guide
  - See **IC 1** for the lower range, and **IC 2** for the higher range. You can ignore parts in IC 2 that involves supporting others (giving feedback to others, giving code reviews etc)
- Khan Academy Leveling Guide
  - See **Beginning Skillful** for the lower range, and **Skillful** for the higher range. You can ignore parts in **Skillful** related to breaking down complex tasks and supporting others

Engineer

- **Alternate names: E4, L4, SDE 2, IC 2**
- **Experience Range: 2-5 years**

The Engineer Level is a large band that ranges from highly productive new grads with a year's worth of experience, to people with 4 years of experience who have shipped significant products.

The theme for this level is **high *individual* productivity**. You ship large projects and features, but don't yet own responsibility for your team's outcomes.

Technical Expectations You can own large features, which can require more than a month or so of work. You are able to break up the project into many tasks, and complete them with little guidance.

You are *highly* effective as a coder. You consistently write code that's cohesive and maintenable. You proactively own and fix the codebase, whether it's upgrading to modern frameworks, adding tests, or refactoring confusing pieces. You demonstrate *high* throughput – within your team, and perhaps the org, you may have one of the highest number of commits or the highest number of features shipped.

People Expectations You give great code reviews to peers and junior engineers. You are a great mentor on the craft of engineering. You have learned how the organization works – from how products get shipped, to what to do if something breaks. You know how to work well with PMs, managers, and

tech leads. You can use that knowledge to join any large initiative and produce quickly. People are often very thankful when you join a project.

More Data Points

- Patreon Leveling Guide
  - **IC 2**
- Khan Academy Leveling Guide
  - **Skillful**

Senior Engineer

- **Alternate names: E5, L5, SDE 3, IC 3**
- **Experience Range: 5+ years**

The Senior Engineer Level is the terminal band for most engineers. This means that at this level, you are are a capable, strong engineering leader, and can stay at this level for the rest of your career if you so choose.

The theme for this level is **team leadership**. You are a strong, saught-out technical leader, and a mentor amongst your team. The most key separator for the senior engineer from the levels below is a deeper expectation on the people dimension.

Technical Expectations You own initiatives that can take six months to a year to complete and involves multiple people. You have deep, sought-after expertise in a technical domain and have a track record of leader large projects well. You can take large problems with tight deadlines, work cross-functionally across teams, create specs, divide up the work, and drive your team to deliver. You have demonstrated this multiple times.

You own the success of a significant system. You often find ways to improve that infrastructure, and build out initiatives that get the whole team on board. The work you do often influences other teams to improve their product as well.

People Expectations As a technical leader of the team, you are a strong partner to your Engineering Manager and a mentor for others. You may take on aspects of engineering management or product management, from leading standups to communicating progress, to hiring. You know how to work throughout different organizations to align and unblock large projects, and have done that multipl times.

You work with your engineering manager to make sure people are developing well on your team, and often step in to mentor some of them to improve. You constantly work to make people around your productive and content – from introducing new best practices, to making development more ergonomic.

More Data Points

- Patreon Leveling Guide
  - **IC 3**
- Khan Academy Leveling Guide
  - **Super Skillful**

Staff Engineer

- **Alternate names: E6, L6, SDE 4, IC 4**
- **Experience Range: 7+ years**

The Staff Engineer Level usually makes up less then 10% of the engineering team. The theme for this level is **organizational leadership**. You are a technical lead that takes over ownership over a suite of systems, and influences organization you are a part of. The key separator for the staff engineer from the levels below is scope – from team ownership to org ownership.

Technical Expectations You own a suite of systems, and drive long-term initiatives to keep them impactful and maintainable. You can break down large problems into multiple systems that a team could work on. You often detangle and work on the hardest parts of these systems. You own problems that often involved coordinated, phased changes to different systems.

You have experience taking systems from conception to production – from idea, to buy-in, to specs, to alignment, to comms, legal, and so forth. You have excellent communication skills, and can navigate throughout multiple orgs to find alignment and launch.

You influence the engineering direction of your org. You are aware of the trends in the industry, and work proactively to keep your org well positioned.

People Expectations You can mentor senior engineers, and are a strong partner to multiple leads. You are a culture vector for your teams, demonstrating the kinds of values that make an exceptional team member.

You champion the people around you, and constantly work to make the whole org more effective. You take part in re-org discussions, take ownership over hiring, and the larger direction for the org. You look ways to multiply your impact – this can involve writing, introducing new tooling, to setting up brown-bags.

More Data Points

- Patreon Leveling Guide
  - **IC 4** to **IC 5**
- Khan Academy Leveling Guide

- **Ludicrously Skillful** –> ## Communicating Signal Now you speak the language of scope & signal and you've formed a conviction about your level.

Now, it's time to **calibrate.** Our goal here is to ensure the signal you convey matches the level you want. See the homework below. As you go through the different interview types, you'll refine more and more the kind of signal you want to send in each type. For now, let's get the narrative and resume right.

Once you have formed your mental game and enriched your communication you're ready to start the interview process strong!

Homework

- Review signal and scope, the leveling guide, and your leveling homework.
- Review example narrative and annotated resume document
- Update your narrative and resume with sentences that demonstrate your signal

Interview Phases

Interview Process Overview

A full loop with a company will look something like this:

- Recruiter chat
- Technical phone screen
- Onsite loop
- Offer stage

Every company has a slightly different process. Some places may have you do take-homes instead of or in addition to a phone screen. Some places may have you talk with a hiring manager before coming onsite. Some places may even expedite you and take you straight to an onsite after speaking with a recruiter. The above is the general flow we encountered based on 20+ loops with companies in San Francisco and the Bay Area. In greater detail:

Recruiter chat

This interview is typically a 15-30 minute call with a recruiter to discuss your interest in the company. Before talking with any recruiters you should already have clarity on your goals for the job search. Your objective for this chat is to communicate those goals, determine if there is a potential mutual fit, and move forward to the technical screen.

Technical phone screen

This interview is typically a 60m minute video call with an engineer where you share your screen and code live. You will work on either an algo or UI problem depending on your interview loop (back-end or front-end). You should clarify with your recruiter what to expect. Before doing these interviews you should be thoroughly prepared for the Algorithm and/or UI interviews. Your objective for this interview is to demonstrate your technical ability, ask insightful questions to your interviewer after the coding portion, and move forward to the onsite.

Onsite loop

The onsite loop typically lasts between 4-7 hours and consists of multiple interviews at the company itself. You will encounter algorithm, UI, system design, and experience interviews. Be ready to meet a lot of people, communicate

enthusiasm, and demonstrate your technical ability. Your objective for the onsite is to give strong positive signal and move forward to the offer stage.

Offer stage

You've made it this far, congratulations! There are no more interviews for you to pass. Your objective at this stage is to determine whether this is the right opportunity for you, and if yes, to negotiate the best package that makes you happy and excited to sign.

Interview Process: Ramp Up

Now that we have our foundations, the next steps are connecting with target companies. Our goals for this section are the following:

- Identify a list of target companies based on narrative
- Craft a resume which communicates relevant signal for your target level
- Start the interview process at target companies through referrals, reach-outs, or direct applications

In addition we recommend you create two more things:

Status document (external)

This is where you list the companies in your funnel, what stage you are at with each, and your current todos. We recommend you use something like Dropbox Paper or Google Docs so you can share this document with your community. This will help you stay on top of things and provide visibility to your team. It can be intimidating to share, but the support you receive will be worth it.

Progress document (internal)

This is where you record your progress throughout the job search. It will include links to your code solutions, performance on interviews, and impressions of companies. Keeping notes about your impressions will be helpful in the choosing process. Furthermore, building a bank of practice questions, real questions, and personal observations will be useful for future interview cycles. At the end of your job search you can synthesize this document into a list of take-aways.

Let's dive in deeper.

Identify Target Companies

Based on your **narrative** defined earlier, come up with a list of 10-15 companies you want to meet. There will be some companies that are more interesting to you than others. Bucket the list into two groups – back-ups and top choices. Going forward, schedule interviews such that you interview with your back-ups first and then your top choices. This way you can get some real practice before speaking with your dream companies. Real practice is useful in all stages:

- Recruiter calls: You will be comfortable explaining your expectations and dealing with push-back
- Technical screens: You will be acclimated to the live interview jitters and have time to correct issues from earlier screens

- Onsites: You will be used to the full-day loop and polished any/all weak spots in different interview types
- Offer stage: By having offer talks with back-up companies first, you will be prepped for push-back and able to respond in a firm yet classy way.

Timing is key.

Craft a resume

Your resume can play a huge role in determining your level. If you have 4-5 years experience you may be a borderline candidate for L4/L5. If you want the senior level, your resume needs to convey L5 signal. This will help ensure your recruiter designs an L5 interview loop for you. Similarly, this will make your case stronger for the hiring committee. Refer back to **Signal** and **Leveling** and see some example resumes attached in this module.

Get noticed

Referrals are the best way to get into the recruiting process. Most companies want and prioritize referrals – so much so that they offer 5-10k bonuses to employees. Identify people you know from your target list of companies and reach out. They will most likely be happy to help. If you're unsure of who you know, go to LinkedIn and search for people who work at your target companies – LinkedIn will show you closest connections first.

If you don't know anyone currently at the company, look at 2nd connections who work at the company and the mutual friends you two share. You can ask your friend how close their relationship is with that person. If they have a good relationship, your friend can pass along your resume to them. This is another good way in.

If you absolutely can't think of a way to get a referral, feel free to send in a direct application. You can certainly get noticed this way, In our experience referrals are truly the best way to go. Of the 20+ onsites we did, every single loop was initiated through a referral of some kind.

Homework

- Create a list of 10-15 companies you want to meet. Bucket this list into two groups: back-ups and top choices
- Craft a resume conveying signal for your target level. Have friends look over it.
- Reach out to your network and get referrals. Send direct applications as a last resort

- Create an external status document and an internal progress document for tracking your job search.

Recruiter Screens

The recruiter screen is usually the first conversation you have with a company. These interviews are rarely technical and meant to get you and the company acquainted. You are pretty much guaranteed to go to the next stage so no need to worry about "failing" this interview.

Your goal is to clearly communicate your narrative and expectations. For example, if you're expecting to be leveled as an L6, it is critical you let your recruiter know early on. They may push back or tell you that leveling is determined post interview – that's fine. Be firm with your expectations, make it clear you are open to interviewing and explain the **scope** you are looking for so your recruiter understands why you want this level. It will help your recruiter build the right interview loop and the right case for the hiring committee.

Communication

Remember to always be positive and classy. Treat your recruiter as a friend. Even though this isn't true – realize they have a vested interest in you succeeding and signing an offer. Assume positive intent. Not all recruiters are created equally and some are better than others. Treat everyone with respect at all times. Make your recruiter part of your team and work with them to get the package you want.

Expected compensation

A recruiter may ask at this stage what your expected compensation is. To get an idea of what you should target, we recommend you look at levels.fyi – However, We personally recommend you do not give any numbers this early. Instead, focus the discussion on determining mutual fit and leveling. It's better to discuss numbers at the offer stage – your goal for now is to make sure you actually like the company and get leveled correctly. If a recruiter keeps pressing, you can tell them a range at the top of your level – emphasize that you believe the company is competitive and you are not worried about reaching a mutual agreement. We highly recommend you look up patio11's salary negotiation article as well as additional negotiation resources to prepare for this scenario.

Homework

- Read patio11's salaray negotiation article

Technical Screen

A typical technical screen will be a 60 minute chat with an engineer. You will most likely use some online collaborative IDE like Coderpad, Codepen, HackerRank, etc. The crux of this interview is to complete the technical question. Although the interview is scheduled for an hour, you should only expect 45 minutes to solve the question. In general, for any technical interview you should estimate five minutes for introductions, five minutes for problem exploration and/or setup, and five minutes post-question to ask your interviewer questions about the company. That means if a technical interview is scheduled for 40 minutes you should aim to solve all the problems/parts (expect a problem to have multiple parts) in 25 minutes or less. This means you need to move **fast**.

For generalist/back-end roles you will most likely be asked an algorithm question. For front-end/UI roles you will likely be asked either an algorithm or UI question. Always clarify with your recruiter on what you should expect. For more information on interview types and success see **Algorithm Interviews** and **UI Interviews**

A word about introductions and post-interview questions. Although they play a minor role, both are still important. For introductions, prepare a solid two to three minute summary about your self, your experience, and your career goals. Keep in mind your narrative, the signal you convey, and your target level. For follow-up questions, prepare a list of purposeful questions you want to discuss. Your goal is to get to know the company and gather info on whether this is the right place for you. Ask what you truly want to know. The worst thing to do at this point is to say you have no questions. This communicates a lack of interest on your part. Even if you nailed the technical part, if it seems like you don't care about the opportunity you may not be passed onto the next round.

This pattern of intro, technical questions, and follow-up questions will repeat for all technical interviews, including those during the onsite. After you've done a couple live interviews you will be adjusted to the format. Post-interview, we strongly recommend you log your performance in your internal **Progress** document. Evaluate your performance, identify what went well and what could be done better. This is crucial for you to improve. You should be confident and near flawless by the time you screen with your top companies.

Timing-wise, We recommend you do all your screens over a two to three week period with at most one screen a day at first and potentially two to three screens in a day later on once you feel confident in your performance. Schedule your screens so that you interview with your top choices last.

If you pass the technical screen the next step is the onsite!

Onsites

The onsite is typically the last stage with a company in the interviewing process. If you do well here you should expect to receive the offer. You should expect to spend between 4-7 hours at the office. You will meet a lot of people and be asked a lot of questions. There are several types of interviews you will encounter that day:

- Algorithm: Similar format and difficulty as the technical screen. See the section on **Algorithm Interview**
- UI: For front-end/UI engineers. Also similar format and difficulty as the technical screen. See the section on **Algorithim Interview**
- System Design: This interview is especially important for leveling. It's crucial you clarify the objective for this interview. See the section on **System Design**
- Experience: This interview is also used for leveling. See the section on **Experience Interview**
- Hiring Manager: In this interview you will hopefully meet your prospective manager. In some places the hiring manager interview will be identical to the experience interview. In any case, make sure your answers convey signal and scope for your target level. This is also your opportunity to get a feel for your manager. Evaluate whether they would be a good fit for you.
- Wrap-up: Your last interviews for the day will most likely be a wrap-up and walk-out with the recruiter. They will usually ask you how it went. Feel free to be honest but always maintain a positive attitude. Make sure to inquire about next steps, this communicates your interest in getting the offer.

Having a positive attitude throughout the day is essential. You want to come in radiating confidence and warmth. Not only will interviewers evaluate your technical ability, they will also evaluate whether they want to work with you. If you find yourself faltering during an interview, keep pushing forward. Never give up. Always maintain a can-do attitude if the interview goes poorly. Stay upbeat. Do not bring self-doubt or negativity into the next interview. One stellar performance can make up for a poor interview. Always be on, be confident, and be positive.

After the onsite, update your external status and internal progress documents. Similar to the screen stage, identify what went well and what you can do better. Incorporate these learning into your next onsites. If you find yourself performing very poorly reschedule your later onsites. You want to be confident by the time you do your onsites with your top companies.

Timing-wise, we recommend you batch all your onsites over a two to three week period. You should start your onsites after you've completed all your screens. Two to three onsites per week is a good cadence. If confident, feel free to go up

to four a week. Beware of burnout though – onsites take a lot of energy. You want to ensure you are upbeat and energetic for each one. If possible, try to have at least one day in-between onsites to rest and reflect.

After you've completed all your onsites it's onto choosing where you want to work and ultimately negotiating the right package!

Offer Stage

Choosing

After your onsites you may find yourself in the fortunate yet uncertain position of choosing between multiple companies. This can be a tough decision. Here are some tips we used to decide:

- Get centered. Go back to your narrative and ask yourself what is most important to you. What are your goals? How does each company fit with your desired path.
- Meet your team. If you haven't already, meet your prospective manager. Ask your recruiter to set up in-person or phone meetings. Your manager has a huge impact on your experience at work. Feel them out, ask yourself if they are the right manager for you. Similarly, if you haven't met other engineers on the team, set up meetings to chat with them about work. Don't be shy about setting up multiple meetings. We visited our top choices several times post-onsite before making our final decision. If they've extended an offer, the company wants you. Take your time in figuring out what is best for you
- Reach out to your community. Bounce your thoughts off your family, friends, and mentors. They may illuminate something you didn't see before. If multiple people are saying the same thing, there may be something to it. That said, if everyone says you should choose company A but in your heart you know you should choose company B – the simple process of talking out loud with people you love and respect will provide clarity.
- Create a choosing document. After going through the above, create a document where you summarize all pros, cons, and thoughts. Writing things down will force you to synthesize everything and determine why a company is right for you.

This may seem like a lot of extra work. In some ways it is. There is no need to do the above if the choice is already clear to you. However, if you are unsure, take your time – it's worth it.

Negotiating

In our opinion the best negotiations are short and concise. We recommend to do this over email. Ask yourself what package would make you sign immediately and send over those numbers to your recruiter. Your email could look something like this:

```
Hi Recruiter,

I'm ready to make my decision. The choice is between you and
competing offer. If we can get the following package I will be
ready to sign immediately.

Base: Number
Equity: Number
Signing Bonus: Number

I'm excited :)

Cheers,
Your Name
```

Your recruiter will most likely give some push-back. They may ask what are your competing offers. They may say they can't meet those expectations. Recruiters deal with lots of candidates and have significantly more negotiating experience than you. By negotiating over email you will be able to thoroughly think about your responses and not bring in your own emotions (negotiating is stressful!) This will significantly improve your odds of getting a better package. The following will also help your negotiations:

Know your band

Do your research on the company and their ranges for your level. Levels.fyi is a good resource for this. If you have friends or mentors at the company, ask them as well. Get an idea of what is realistic and evaluate whether your target compensation is in band for your target level

Have multiple offers

This is one of the strongest forms of leverage you have. If you have offers from other competitive companies you should certainly let your recruiter know. This will allow them to make a strong case to the compensation team to approve your package. Even if the other offers are lower they may bump up your package to close you

Be willing to walk away

This can be very hard to do, but ultimately one of the most powerful things you can do for negotiations. It is also extreme and should be a last resort. Do not do this unless you are truly centered on walking away if your expectations are not met. Suppose you have an offer from your dream company which is lower than an offer from another good company. In this scenario dream company won't budge – assume this is final. After factoring everything in, are you truly willing to walk away? If instead your expectations were met would you sign immediately? If you answered yes to both questions, let your recruiter know that you can't take the offer unless your package is bumped up. At this point feel free to hop on a call, since you're now centered on your decision you will be free from any fear. Your resolve will communicate strength. The ball is entirely in their court.

This strategy is not limited to just compensation. Suppose you were leveled lower at dream company vs good company but overall compensation was comparable (or even higher at dream company). However you evaluate that the expectations, responsibilities, and growth at good company would be higher. If those things are important enough to you, ask yourself whether you are willing to walk away from dream company. If so, let your recruiter know that without the higher level you cannot accept.

For more tips we recommend you loop up pattio11's salary negotiation article as well as other negotiation resources. An hour or two of research could you net you an additional $100,000+ – definitely worth your time!

A word on exploding offers

This is an unfortunate tactic used to close a candidate and prey on their fear that the offer may fall through. Hiring good people is hard. The company has already heavily invested in you by having you go through their process. They will not rescind if you need more time. Be polite but firm with your recruiter about your needs and expectations.

Once you have negotiated a package that you are willing to sign, ask your recruiter to send over the offer. The offer letter will most likely have a deadline – but this is reasonable since you are ready to sign immediately.

If you've made it this far and signed – congratulations! An exciting new adventure awaits you. All that is left on your part is concluding / sharing the news with your community.

Concluding

Congratulations for making it this far. Hopefully you feel some relief after signing the offer. It's been a long journey, and we hope you are excited about your next opportunity. Although your job search is done, we recommend you do a couple more things to conclude well.

Inform other companies

If you had other offers, let your recruiters know. Email is fine, but letting them know over the phone is really classy.

Update your community

Now that you've signed let your people know! They will be excited to hear the good news.

Send thank-yous

You should send these to people you deeply connected with throughout the process. At a minimum we recommend you send a note to your future manager and teammates if you met them. They will be happy to hear that you joined.

Send LinkedIn Requests

The valley is small, just because you turned down an offer doesn't mean you won't connect with a company or manager in the future. Throughout your job search you've met a lot of talented people. This is a great opportunity to expand your network. Who knows where you'll end up next.

After doing all this you've beautifully concluded the job search. Well done!

Algorithm Interview

Algorithm Interview Foundations

The algorithm interview is typically a 60 minute chat with an engineer in which you solve a technical problem. The expectation is to have a working solution before the end of the interview, with time left over to ask the interviewer questions about the company. The beginning of the interview will include introductions and problem explanation as well. You should aim to solve the problem and all of its parts within 30 minutes. There is a general framework for how to solve the problem which involves:

- Ask clarifying questions
- Come up with an approach, without writing any code yet
- Validate the approach with the recruiter
- Implement the solution with code
- Demonstrate correctness by running through test cases

You want to leave the following impressions with your interviewer by the end of the interview.

- You are a strong problem solver
- You are a great communicator
- You are thorough and write great code
- You move fast

Doing all the above sends great signal and will earn you a strong yes.

Let's break down this interview.

The bar

Here are some example problems that mirror the difficulty level of what you can expect in an actual algorithm interview. This is not an exhaustive list, only meant to give you a rough idea. You should feel confident handling these kinds of questions by the time you do actual screens and onsites. Your goal is to have a working solution that compiles and passes test cases within 30 minutes or less. All these questions can be found by searching their name on LeetCode.com – in general this is a useful resource for practice questions.

- Merge Intervals
- Meeting Rooms
- Number of Islands
- Spiral Matrix
- Coin Change

- Parse Lisp Expression
- Implement LRU Cache
- Text Justification
- Wildcard Matching
- Minimum Window Substring

You can also get a preview of the questions a company asks by doing a google search for "company interview questions glassdoor" or "company interview questions github". Sometimes you may find actual interview questions that will be asked on the screen or onsite!

Framework

The first step is to learn the problem solving framework. One of the best resources for this is Cracking the Code Interview (CTCI). We recommend you read chapters I - VII. These chapters thoroughly break down the framework for solving the algorithm interview. Internalize these chapters and apply this methodology when solving any algorithm question.

Foundations

With the framework in mind, the next step is to do a lot of practice problems. We recommend you do questions from either CTCI or Elements of Programming Interviews (EPI). We think these are the two best resources for algorithm questions. It is highly unlikely you will encounter a real interview question that isn't covered by these two books. We have a slight preference for EPI since it covers more questions than CTCI. At a minimum, you should do all problems covering the following sections:

- Strings
- Arrays
- Linked Lists
- Stacks / Queues
- Heaps
- Trees
- Hash Tables
- Searching
- Sorting
- Recursion

Once you have strong foundations in the above, reach further and do problems covering:

- Dynamic Programming
- Greedy Algorithms
- Graphs

- Bitwise Operators

These are really hard questions and you may never be asked them in your process. You will be extremely well prepared if you put in the time to solve and understand these kinds of problems.

When doing these problems, set a timer for 20 minutes and try to solve each question on your own. Look at the solution if you finish or time runs out – spend as much time as you need to understand the answer. There are a lot of tricks and concepts to learn, take the time to internalize them. Once you know the trick, future problems will be easier. Your goal isn't to see and memorize every problem / solution. Instead, your focus should be to increase your bag of tricks and sharpen your problem solving abilities. Follow the pattern of coming up with an approach, implementing it, and running your code against test cases.

By going through lots of problems in different areas you will be ready to take on anything.

Homework

- Read chapters I-VII of CTCI
- Get your foundations by reading the following chapters in either CTCI or EPI
  - Strings
  - Arrays
  - Linked Lists
  - Stacks / Queues
  - Heaps
  - Trees
  - Hash Tables
  - Searching
  - Sorting
  - Recursion
  - Dynamic Programming
  - Greedy Algorithms
  - Graphs
  - Bitwise Operators
- Solve sample questions that match the interviewing bar difficulty by searching the following on LeetCode.
  - Merge Intervals
  - Meeting Rooms
  - Number of Islands
  - Spiral Matrix
  - Coin Change
  - Parse Lisp Expression
  - Implement LRU Cache

– Text Justification
– Wildcard Matching
– Minimum Window Substring

Mock and Real Algorithm Interviews

Once you have your foundations it's time to do some mock interviews. There is a difference between solving problems on your own vs in front of someone else. For example:

- Throughout the interview you need to communicate your thought process out loud, verbalize time/space complexity, and collaborate with the interviewer.
- For screens, you may be required to write code in an online collaborative IDE (Coderpad, Codepen, HackerRank, etc)
- For onsites, you may be required to write code on the whiteboard

By the time you do the real interview you want to be comfortable with all the above. As a result, we strongly recommend you do mock interviews before your first real algorithm interview. Here are some great resources for mock interviews:

- Pramp.com – Pramp is an online platform where people mock interview each other. Each interview is 30 minutes and the questions are about medium difficulty level. You code in an online IDE so it's a great way to get used to that if you've never used one before.
- Interviewing.io – Do these once you find yourself easily passing Pramp. These interviews are 60 minutes long and the questions are typically harder. The format is almost identical to a real technical screen, so this is excellent practice. The interviewers are paid so you expect a more professional tone compared to Pramp. We strongly recommend you do at least one of these before a real interview
- Friends – Ask your engineering friends to interview you. If possible, get access to a whiteboard. Whiteboard-ing can be stressful, you certainly want to get some practice solving problems on the board in front of someone.

As you go through these practice interviews evaluate your performance and identify areas to improve. As you start feeling confident, go ahead and schedule the real interviews.

Real interviews

There should be no surprises during the real interview if you've put in the time with CTCI/EPI and done several mock interviews. You may feel some jitters in your first screens/onsites, but this is normal and will go away. Push through and maintain a can-do attitude – you've got this! Remember to log your performance after each interview. Ideally time your interviews so your top choices are last. This way if you uncover any more gaps you can correct them before meeting with your dream company.

Conclusion

You should now have an idea of where the bar is at and how to get there. It may seem like a lot of work to solve contrived puzzles that ambiguously correlate to technical ability. This may be true – but this mindset won't help you succeed in this interview. Instead, we hope you find some excitement in solving these algorithm problems. The algorithm interview is like a game. The objectives are clearly defined and all the preparation material is available beforehand. It's up to you to put in the time to win it. Study hard and have fun!

Homework

Do these after completing the homework from the previous section on foundations.

- Complete at least 1-2 pramp interviews
- After successfully passing pramp interviews, complete at least 1-2 interviewing.io interviews
- Complete at least 1-2 practice interviews with friends

If you're doing well in your practice interviews you should be ready for your real technical screens!

UI Interview

UI Interview

Let's go over the UI Engineering Interview.

We think this is one of the most fun interviews, because it's the interview that's most relevant to your day to day work as an UI engineer.

When you go into the UI interview, you'll work together with an engineer over a live session — usually over Coderpad or Codepen. You'll encounter two kinds of problems:

Either you'll work together to build a UI — perhaps a modal, autocomplete, drag drop, etc. Or, you'll build a javascript library — perhaps a bunch of lodash functions, or a pub-sub system that takes advantage of some core javascript principles

Through the questions, the interviewer is looking to parse two signals:

- Javascript Experience. They want to know, can you actually build out UIs and be productive at your job? Are you familiar with the practices in the industry, for building client-side applications?
- Engineering Fundamentals. They want to know, do you know more than just javascript? Are you familiar with data structures, can you model data well, can you communicate your ideas, and would you be able to build an end-to-end system?

To communicate this signal the best, we suggest you get comfortable with the following:

Setting up online editors.

You'll constantly be working in Codepen. You'll most likely want to use the libraries you use at work — usually React — and you should be able to set up an environment that renders "hello world", in a few seconds.

Building a bunch of UIs from scratch in online editors.

You have been building UIs at work all the time, but it could have been a while since you have built a modal from scratch, or autocomplete, or worked with drag and drop. In the homework section, we've included about 4 or 5 different UIs that we suggest you build. These are meant to remind you of some of the JS concepts that you may not have gotten to do frequently at work. These

questions are also **very frequently** asked in real interviews. Although some examples, using JS's selection api, or drag and drop, are uncommon, building a few projects here will ramp you up very quickly.

Debugging in online editors.

You may already have mastered chrome's developer tools from work, but it always helps to brush up. It'll also be a bit different in online editors. You should be able to freely use debugger and console.logs to quickly narrow down on bugs. We've included an article that goes over all the tools you can use in Google Chrome. This can be helpful in and out of the interview. As you practice building a bunch of UIs, make an effort to use the debugging tools to narrow down your bugs more and more quickly.

Latest and greatest in JS.

The example UIs may not specifically cover what you know and don't know. We want you to reflect — what are the concepts that you've wanted to learn, that you haven't learned yet? Perhaps you've rarely used Canvas, or haven't tried Redux yet, or want to play with service workers. Catching up on these will help send **very positive** signal during the interview — showing you truly love the industry and are a self-learner. We suggest taking a look at some of Wes Bos's courses — he loves the UI world and goes deep on a bunch of the latest concepts.

Data structures and algorithms

Even though you are in the UI engineering loop, and the chance you get asked heavy algorithm questions is lower, we strongly suggest preparing for algorithms anyway. Go through the **Algorithm Interview** section, and prepare for this interview too. If you prepare well for those, most of the coding-related questions in the UI loops will feel easy.

The live interview

When you go through the interview, we suggest you structure your answers like this:

- Write the JS logic, and do not worry about CSS.
- Do not worry about breaking things up into too many different functions or components. This doesn't necessarily send signal, and if the interviewer has a follow-up that changes the question, you'll end up having trouble refactoring.

- Once a js implementation is done, **verify that it all works.** As you verify, communicate what you are checking, and how you are checking it to the interviewer. This will help you catch a bunch of edge cases, sending positive signal to the interviewer.
- If you end up having time, you can start abstracting, or just mention how you would abstract

To get good at doing all this, we suggest you schedule 2-3 practice interviews with your UI engineering friends, or use Pramp. If you follow this, we're confident you will succeed at this interview and grow as a UI engineer!

Homework

1. Set up Codepen with your environment. Go to codepen.io, and create a new pen. Try to set up the environment that you normally use (For example — React + Babel). You can do this by clicking on Settings, and adding the right library. Your goal is to render "hello world"
2. Get comfortable building UIs. Here are sample problems to work through
   - Create Tic-Tac-Toe
   - Create a Trello Board
     - Allow creating boards
     - Allow creating cards
     - Allow editing cards
     - Allow moving cards
     - **Bonus: Allow drag and drop**
   - Draw a sudoku board. This is a 9x9 grid, with each 3x3 square colored differently. Part two, given a 2D array with sudoku solution, render the board with the values and highlight any invalid rows, columns, or squares.
   - Create a Credit Card Input. As you enter the numbers, it should add spaces after every 4th digit
   - Create an event emitter. Think about what is the right data structures to use here? All of your calls — subscribe, unsubscribe should be O(1). See demo code below for expected behavior
   - Create a popover component
   - If you're comfortable with these, ask yourself — what do you want to learn? Create a quick project for that
3. Get good at debugging
4. Catch up on the latest and greatest of JS
   - Wes Bos's courses are great — search up on youtube anything you're curious about — search up talks
   - Check out some interesting articles and start to go deep from there
5. Schedule 2-3 mock interviews with friends to do over Codepen

Event Emitter Demo Code

```
const emitter = new EventEmitter();
const subOne = emitter.subscribe('foo', fnOne);
const subTwo = emitter.subscribe('foo', fnTwo);
const subThree = emitter.subscribe('bar', fnTwo);

emitter.emit('foo', 1, 2, 3);
// fnOne(1, 2, 3), fnTwo(1, 2, 3) is called

subTwo.unsubscribe()
emitter.emit('foo', 1, 2, 3); // fnTwo is removed
// fnOne(1, 2, 3)
```

System Design Interview

Before the System Design interview

In a system design interview, you'll meet a senior engineering leader, who's going to ask you an open-ended question: It can be something like "Let's create a feed for Twitter", or "Let's implement a Hotel Booking System".

Your goal will be to grab the helm for the next 45 minutes, crystallize the problem, and architect a system that solves it. This will demonstrate your ability and experience in your technical leadership.

This in our opinion, is one of the scariest interview types, but also one of the most fun. It's scary because of how opened ended the question is, but it's fun because you get to solve a big problem. You'll learn how some of the most complicated and interesting systems are built — from Google Photos to Uber. The best part? All this learning will transfer from the interview room, into making you a better architect.

Mastering this interview

First, and foremost, come in with the right mindset. Most people come in with the mindset that this is an exercise — it's like they are "playing house", but for building systems. You can see this when candidates say things like they "would not do in real life." They may ignore some of the bigger technical problems they see — this is after all an exercise.

Unfortunately this will send the wrong signal — the interviewer will think you didn't see the big problem.

Let's change this mindset, instead of playing house, let's build a real house. When you're about to start the interview, take a moment, and imagine you are at an **actual meeting at work.** In that meeting you are told — here's this problem, it is the highest priority now, and we need to get a team on this ASAP. Can we brainstorm the roadmap?

Your goal at the end of 45 minutes is to **have that roadmap** — something that can be divided up between a team, with open todos and a plan forward.

When you mindset is I am solving a **real problem** the rest of the interview will start off strong.

System Design Interview: The Functional Spec

The question you get asked will be ambiguous. Your goal for the first phase of the interview is to take that problem, and make it concrete.

To show great signal here, you'll want to demonstrate leadership by taking initiative and leading the discussion. With very little prodding, you should be able to flesh out all the requirements for your system.

Here's how you can do that.

Step up to the whiteboard, and write down these words

- Not Dos
- Functional Spec
- Technical Spec
- Related Problems

Then, imagine you need to deliver this **for real**, and start asking question:

First, explore the constraints — if we are building a Twitter feed — are we building for iphone and android too? Do we support offline mode? Do we support privacy settings?

Then, explore the functional specifications — can we have retweets, do retweets come with comments, do we support deleting tweets?

As you go through, add notes in "Not Dos", and "Functional Spec". If a similar system comes to mind that you would research if you had more time, note it down in "Related Problems"

As you do this, actually draw out, every screen that we will support in our implementation.

For example, for feed, we would

- Draw the home page, including the login box, and a note on what would happen if the user is already logged in
- Draw the screen for the profile, showing the profile info, the list of feeds, how retweets look like

Work with the interviewer here — you'll end up finding a lot of "not dos", and get a much better sense of the system we will need to implement, and the tradeoffs that will come as a result.

By the end of this, you will have all the functionality you need to support. Most people get stuck in this interview or jump into coding too quickly, so if you take the time here to plan and come up with a functional specification, you are showing very positive signal to the interviewer.

Once the functional spec is done, you're ready to move to the next step in the interview!

System Design: The Technical Spec

Now that we know much more clearly what problem we're solving, let's go broad, and build a **technical spec.**

Your goal here is to outline all of the component parts we would need to build in our system. Imagine you will divide up work across a bunch of senior engineers, and you want to find out who will own what.

Here's how we suggest you do this.

Look over each screen, and draw out the systems we would need to support everything.

For example, we'll need a web page, so draw a box for that. We'll need an auth api, so draw a server box, and name the APIs. We'll need to store the data, so draw the database box. We may need caching, so you can draw the redis box.

To do this well, we suggest two paths:

If you are already comfortable with distributed systems — you can go right in and start drawing out the complete system — discussing how you would add redundancy and noting down the kinds of tradeoffs you would make.

Alternatively, you can start off by saying explicitly **let's first build a simple system, and scale it from there.** It's important to explicitly mention this to the interviewer, so you aren't interrupted in the middle. Once you draw a complete simple system, start asking over and over — what will fail, and fix them, **noting down the kinds of tradeoffs you are marking.**

Either way, you will have finished this step once you have outlined broadly a complete system, that would solve our problem and scale.

One of the biggest mistakes people make here is ignoring the big problems that come to their mind.

For example, in Twitter's feed API, how would we load new tweets? We would need to handle merging with old tweets, and based on how we do our pagination, we may end up missing tweets.

This kind of problem is almost **the whole purpose of the interview.** Detecting this problem demonstrates your experience to the interviewer, and coming up with the right tradeoff shows your technical skill.

The key realization is this: you don't need to solve these problems at this moment yet. Instead, say the problem that comes to mind, the interviewer will either clarify what they expect, or you can note it down to handle later when you get to the API.

Once you're done with the technical spec, every screen you drew out should be supported by your system.

System Design: Going Deep

Now that we've gone broad, it's time to go **deep**.

At this point in the interview we've noted down a bunch of tradeoffs and problems remaining to be solved. It's time do demonstrate our expertise by going deep.

Here's how we suggest you do this:

Start by looking at the most critical problems — perhaps it's scaling a particular piece, or designing a particular API.

Imagine now that you are giving this piece to a relatively strong, but inexperienced engineer. Now, spend the time to really unravel, and build up the spec for this piece.

For example, say we are building out the twitter feed api — we need to answer what the parameters look like, how will paginating work?

Now we will handle how the client will merge the data. If any tradeoff comes to mind, make sure to mention it. For example, we could have feed connected through websockets, or through http — what would the tradeoffs be for both?

Go problem by problem, until you are satisfied that you've dealt with the most important tradeoffs. If you have time, keep going.

Choose the sections that you are strongest in and start to expand out — For example, if you have experience in machine learning, you can go deeper on the fraud detection for tweets — how would the models be deployed, what would the feedback loop look like, how would that piece scale, what would the API look like?

By going deep, you will have communicated your ability to break down problems, find the tradeoffs, and build a complex system that solves the problem. You would have also communicated real experience and expertise through your deep dives.

Go the extra mile

Your system is complete, but you can go further — what are the steps we would take to ***launch?***

For example — how can you make this testable? How can you make this maintainable? What would the deployment process look like? Do you see any problems for extending the project? What kind of alerts would you set up? What would you have in your oncall document?

This is your opportunity to demonstrate you have **strong competence** in launching projects like this **from start to finish.**

If you are able to get the extra mile, and you take a picture of that whiteboard, you actually will be able to use this to kick off this project.

Congratulations, you've now gone through all the phases of the system design interview!

System Design: Final Advice

In this section we'll discuss how to prepare and give some final advice for the system design interview. Let's get right to it.

Internalize the phases of the interview

Try to schedule 2-3 practice interviews, and note how you do on each of the phases – Functional Spec, Technical Spec, Going Deep. Ask for feedback and evaluate. The more you do this, the more comfortable you'll be driving the interview. However, as you go through this, you'll notice areas you know less well.

Start filling those gaps

Through your practice, you may have noticed, for example, that you don't know concurrency models too well, or maybe you realize you have less experience building real-time systems. Note those areas, and make a plan to improve. In the homework, we've included a few ideas and book recommendations, to get better in some of the key system design areas. Go deeper there, and not only will you improve your interview performance, but you'll truly improve as an architect. With that, you know the steps, and you have a plan to improve.

One more thing

There are many, many pieces here that you may not know well.. Remember it's normal not to know a lot of this. To perform flawlessly here – going all the way to the extra mile – it's expected that you'll have years of experience building systems in production.

Interviewers are **trained to expect** that candidates will need to be prodded and challenged. For example, if you only have a few years of experience, it is **normal** that you will need prodding, and won't know some of the tradeoffs in larger systems. Be kind to yourself throughout this, focus on your strengths, and get excited about learning.

Now you're ready for the system design interview, here's to becoming a great architect!

Homework

1. Schedule at least 2 system design interviews. Reach out to senior engineer friends or mentors who could connect you with senior engineers. Ideally you want someone who has experience interviewing others on system design.

Make sure you get a sense of the interview flow, and ask for feedback. By the end of the practice interviews, you should have a sense of the areas you are weak in.

2. To learn these areas, and make a study plan. Some suggestions
   - Learn more about real large-scale production systems, watch InfoQ videos: Some examples: Pinterest, Dropbox, and Twitter
   - Check out some Rich Hickey Talks
   - To learn about databases, get the book "7 databases in 7 weeks"
   - To learn about concurrency models, to get the book "7 concurrency models in 7 weeks"

Experience Interview

Experience Interview: Purpose

In this section we'll introduce the experience interview and discuss its purpose.

When you walk into the experience interview, you'll meet a senior engineer or a manager. Over the next 45 minutes, they'll ask you a bunch of personal questions — things like:

- Tell me about the kind of person you find hardest to work with
- Tell me about a time you wanted to change something outside of your main responsibilities

Most people prepare the least for this interview. It seems like the questions are random, and with all your other todos, it's hard to make this a priority.

However, this interview can make or break your decision to join.

First, it has a huge effect on determining your level, and from our video on leveling, we know the impact this can have on your career and your compensation.

Second, this interview lets you get a strong sense of leadership values for this company, ultimately that'll make a huge difference in your choice to join or not.

It makes sense to prepare. The good news is we only need a bit of work to have outsized results.

The first key to our preparation is to understand why are they asking us these questions?

The interviewer wants to understand two things:

- Do you have the habits and beliefs to be successful at their company? "Are you a culture fit?"
- At what level of leadership can you contribute? "What level are you?"

If we can communicate clearly in both areas, we will have excelled in this interview.

Experience Interview: Communicate Fit

In order to be successful at a company, there are traits that you learn along your career. Some companies value different traits, but in general, there are a few that apply across the board. For example:

- Motivation — Why are you in engineering, why are you choosing this company specifically?
- Empathy — Do you know how to work with your peers, your managers, your reports?
- Proactivity — Can you get work done, even when you don't have all the information?
- Growth — Are you open to feedback, and constantly iterating towards something new?

The interviewer will try to suss out whether you have these traits. Now, if they asked you directly, you would just say "yes" and there would be no signal.

But, they can ask you indirectly — "tell me a time where you made a design decision that you later regretted?" Based on you answer they can figure out how your view on growth and empathy is — do you think you made mistakes at all? How did you deal with it? How was your communication with others throughout it?

The homework in **The Mental Game** will help you convey yourself well.

If you have a strong narrative, your motivation and proactivity will come through. It isn't possible to prepare for every question, but if you are centered, most of the signal will take care of itself.

Experience Interview: Communicate Leveling

One of your interviewer's goals is to suss out your leadership experience. The main way they do this is by extracting *scope* from your answers.

Suppose you are asked "tell me a time where you made a design decision that you later regretted?" Let's go over some possible answers.

Answer: I made this library, and it became highly coupled. **Scope**: You are communicating that you dealt with libraries — the scope of a new grad

Answer: I owned this part the project, and I under-estimated how long it would take. **Scope**: You are communicating that you dealt with features — the scope of an L4 engineer.

Answer: I owned this project, and made too many tradeoffs to launch — it took us much longer to get back to quality. **Scope**: You are communicating your leadership over a team — the scope of an L5 engineer.

Answer: I owned a suite of services, and spent so much time maintaining them, that I didn't see we could re-architect it like this. **Scope**: You are communicating your leadership across the org — the scope of an L6 engineer

Again, the way this is parsed is indirect, so the key to communicating this signal is your homework in **Communication.** If you have a good sense of the leveling guide (and you did the exercises of writing up examples of what you did for your level as well as the one above it) the stories you tell will naturally communicate leveling.

Experience Interview: Follow-up Questions

We already discussed that the best way to communicate culture is from your homework in Mental Game and the best way to communicate your level is from your homework in Communication.

At the end of the interview, you'll have an opportunity to ask questions — you are speaking to a senior leader — and they could give you excellent context on how it's really like to work at the company.

Ask questions from your list you prepared earlier and really try to find out if you would like to work at this company. This too will communicate signal.

For example, if you really care about having a large scope you can ask — "what if I wanted to start a project that goes past my manager, how is it like doing that?" Your interviewer will gain further signal on your initiative.

Remember though, your goal here is to actually get a sense if you want to work here. Do not try to pretend. Simply own exactly where you are at — pretending will come through. What you're trying to do is to find a job that fits with you, and this is exactly the interview to do that.

Now, you're ready for the experience interview!

Homework

- Ensure you've done your homework in Mental Game and Communication

Takeaways

Here are our top takeaways from the course:

1. Referrals. These are the best way to get noticed and start the process. Direct applications should be your last resort
2. Know where the bar the is at. You should have a clear idea what level you need to be at to pass each interview
3. Study and execute so your performance is at that level
4. Communicate signal and scope. Know your target level and architect your resume, recruiter talks, and experience/hiring manager chats around that
5. Time and batch your interviews so you meet with your top choices last and that all your offers land at the same time
6. Meet with your prospective manager and team before making a final decision. Make sure you choose the right place for you
7. When negotiating, don't give your number first, discuss leveling. If pressed for a number state the upper range for your level and communicate you know the company is competitive
8. For final numbers, email your recruiter the package that will make you sign immediately. Email negotiation is straightforward and concise
9. Always be classy. Even if a recruiter forgets to call you, gives you unexpected news, gives you an exploding offer, etc. Carry yourself with class and be firm with your expectations. This behavior will set you apart and communicate your strength and seniority.
10. Be kind to yourself. Interviewing is hard. Sometimes things won't go as well as you like. Don't beat yourself up. Maintain a positive dialogue with yourself. Do your best and be proud.

We hope you found this course useful. Go forth, get your dream job and an amazing package!