Software Engineer

Full Loop Interview Guide

Meta

What You'll Find in This Guide

Welcome to your preparation guide for your full loop interviews at Meta! Our engineering leaders and recruiters put together this guide so you know what to expect and how to prepare. Use the sidebar to quickly jump to the section you are looking for. You will find an appendix at the end of this guide with specific resources and tools that you may reference and utilize. Remember, your recruiter is there to support you, so please reach out to them with any questions.

Accommodations Process

Before you get started, it's important to note that Meta is committed to providing reasonable support (called accommodations) in our recruiting processes for candidates with disabilities, long term conditions, mental health conditions or sincerely held religious beliefs, or who are neurodivergent or require pregnancy-related support. If you need support, please reach out to accommodations-ext@fb.com or your recruiter.

Interview Process Overview

Your interview process will include 4 to 6 different conversations, each lasting about 45 minutes; each interviewer will leave a few minutes at the end for your questions. Your recruiter will give you more specific information about how many conversations you'll have and what type of prep you should focus on. You can expect the following interviews as part of your full loop round:

- Coding
- Design
- Behavioral

Video Conference interview best practices

Your recruiter will let you know whether your full loop interviews will be conducted over video call, or in person. If your interviews will be conducted by video call, follow these tips to set yourself up for success:

- Most important: if you'll be interviewing remotely, make sure you've practiced using the interviewing tools like: Excalidraw and Coderpad.io
- Make sure you're in a quiet environment.
- Double check that you have a reliable internet/phone connection.
- It's okay to ask the person you're speaking with to speak slowly if you can't catch what they're saying.
- You'll need a laptop with a webcam, speaker, and mic. We recommend using a headset or headphones with a mic for better quality audio, but this is optional.

Interview Dress Code

As you're probably aware, we promote a casual environment at Meta so that everyone can be their authentic selves. Formal dress is not required (jeans are definitely ok!). Dress comfortably. We care about what you can do, not what you wear.

Coding Interview

What can you expect?

The interview focuses heavily on coding. You'll be assess on how you solved the problem as well as the structure and style of your code. It's best to avoid bugs, but the interviewer will not compile your code so don't worry about making minor mistakes. Finding and catching bugs in your code is a positive sign!

Additionally, they'll want to hear your thought process throughout, so be sure to provide a narrative as you go through the code. You're welcome to code in whatever language you feel most comfortable, but choosing one that is going to assist in getting an optimal solution in the most speedy and efficient manner is key. Please confirm the coding language preference with your recruiter.

What do we look for?

Your interviewer will be thinking about how your skills and experience might help Meta as well as how you tackle problems you're not as familiar with. In your coding interview, your interviewer will assess your performance on four areas:

- **Communication**: Are you asking for requirements and clarity when necessary,

or are you just diving into the code? Your coding interview should be a conversation, so don't forget to ask questions.

- **Problem solving**: We're evaluating how you comprehend and explain complex ideas. Are you providing the reasoning behind a particular solution? Developing and comparing multiple solutions? Using appropriate data structures? Speaking about space and time complexity? Optimizing your solution?
- **Coding**: Can you convert solutions to executable code? Is the code organized and does it capture the right logical structure?
- **Verification**: Are you considering a reasonable number of test cases or coming up with a good argument for why your code is correct? If your solution has bugs, are you able to walk through your own logic to find them and explain what the code is doing?

How to prep

Interviewers can only assess your skills and abilities based on what you show them during your interview, so it's important to plan and prepare to best showcase your strengths.

As you begin preparing, please reference your Career Profile for additional role-specific prep materials. We use many different coding languages at Meta so practice in the coding language that you are most confident in. Scripting languages tend to be easier to use in coding interviews, but stick with a language you are comfortable with rather than attempting to learn a new one.

You should expect to solve about two problems in the course of about 40 minutes. When you have a solution, be sure to review it. Make sure that it's correct, that you've considered the edge cases, that it's efficient, and that it clearly reflects the ideas that you're trying to express in your code.

In addition to reviewing the above fundamentals, these tips may be helpful:

- **Schedule time to study and practice.** Block out time every day to write code. You'll need to practice writing code without executing it - without the help of a compiler and debugger. This will simulate a timed interview environment. There are several resources available in the Preparation Hub within your Career Profile including coding puzzles and practice interviews.
- **Prioritize breadth over depth.** It's much better to practice solving fewer example problems of many problem types than to become very familiar with one type at the expense of the others.
- **Think about different algorithms and algorithmic techniques** (e.g., iteration, sorting, divide-and-conquer, recursion). We do not ask dynamic programming questions so don't spend time prepping for that technique and focus your efforts elsewhere.
- **Think about data structures**, particularly the ones used most often. For

instance, array, stack (or queues), hashtable, trees (including specialized trees like binary trees and binary search trees), graphs, and heaps.

- **Analyze the problem and make sure that you fully understand it before jumping in.** It's OK to ask clarifying questions during the interview to ensure you understand the exact problem you are trying to solve. Also, breaking down the problem into smaller pieces can be helpful.
- **Get comfortable with the medium you'll use in the interview**. In other words, try writing code with just a plain text editor with no compiler, linter, syntax highlighter, autocomplete, and so on. Please also practice talking through problem before you start coding. Your interviewer will be evaluating how you explore the problem space and weigh different possible solutions so it's crucial to help the interviewer understand your choices.
- **Keep things simple**. If you find the solution getting excessively complex, step back and ask if there's a simpler way to solve it. It may also be helpful to write everything out so you can see insights and bugs faster and make fewer mistakes.

Design Interview

This interview may be systems or product design focused depending upon the specific position for which you are interviewing. While much of the prep information will be the same across all interview types, please be mindful that there are key differences between the interviews that we will discuss below. Please connect with your recruiter if you are unsure which interview type to prepare for during your prep.

What can you expect?

The design interview is 45-minutes and covers the design of complex systems or products and the tradeoffs within your design. This will be a challenging and deeply technical discussion and the scope of the question can vary widely.

- **Systems Design**: The Systems Design interview is more focused on distributed systems, scalability, performance and efficiency.
- **Product Architectural Design**: The Product Architectural Design interview is more focused on API design, usability and utility.

What do we look for?

The purpose of this interview is to assess your ability to solve a non-trivial engineering design problem. The interviewer is trying to determine if you can architect a solution to a higher-level problem that requires connecting multiple concepts. The question will be something very high-level and it will be up to you to drive the conversation. In this interview, your interviewer will assess your performance on four focus areas:

- **Problem navigation**: Are you demonstrating your ability to organize the

problem space, the constraints, and the potential solutions? You should be asking questions to reduce ambiguity, identify the most critical problems, understand what is needed for quantitative analysis, and define a requirement set to design to.

- **Solution design**: We are evaluating your ability to design a working solution that addresses either the complete problem/design or pieces of the problem/design. Are you able to consider the big picture in your design?
- **Technical excellence**: Can you dive into the technical details when needed? Can you identify and articulate the dependencies and trade-offs? How are you mitigating potential risks and failure points?
- **Technical communication**: Can you articulate your vision and technical ideas clearly? We are assessing your ability to communicate your reasoning as well as understand and address feedback from the interviewer.

How to prep

This may be a challenging interview to prep for given it is so interactive and there's no "right answer" for a design problem. Use the prep tips here and the resources listed in the appendix.

- **Improve upon a design.** Think about and review the complex systems you've already designed. What would you change about your approach if you did it all over again? What worked well?
- **Design from the ground up.** Think about how you'd design a system that Meta (or another large tech company) already has. It's a good exercise to think through the complicated, high-scale systems that you already use every day. How would you design it from scratch? What are the bottlenecks to your solution? How would you address them?
- **Think through tradeoffs.** Avoid focusing on only optimal solutions. We're much more interested in seeing how you think through basic tradeoffs.
- **Do some research**. Read engineering blogs about approaches that have worked for big companies along the way. Read about the false-starts too!
- **Be holistic and detailed.** Practice moving effortlessly from high-level information to the precise details. Outline the high-level requirements by describing what components you may need, how they fit together, and how to get to that solution.
- **Practice:** Find a few system design questions online and try to answer them on paper in roughly 30 minutes per question.
- **Start with the requirements.** Your interviewer might ask: "How would you architect the front-end for a messaging system?" Obviously, this question is extremely vague so you could start with some requirements:
    - How many users are we talking about?
    - What should the experience be while you're waiting for confirmation?

- How will you show error states?
- What are the latency requirements for sender–receiver message delivery?
- What kind of features are we going to need to support?
- What operations does this data store need to support?
- How do you push new messages to clients? Do you push at all, or rely on a pull-based model?

During the interview

System Design

In the system design interview, we will expect you to demonstrate an understanding of the low-level restraints and how they affect the high-level goals. Be sure to talk through your solutions while keeping in mind different tradeoffs. Some areas to discuss in system design:

- **Testability:** formally, or otherwise, gaining confidence that the components are correct.
- **Usability:** customer's experience with the system or product and how to evolve it quickly.
- **Extensibility:** changing the software system over time.
- **Security:** can the system or product survive DDOS, spoofing, tampering, repudiation, etc.
- **Portability:** execution in different environments.
- **Availability:** how the system survives failures.
- **Scalability:** how the system can grow over time.
- **Operational characteristics:** diagnosing or debugging problems when they occur.

Example system design questions:

- Design a URL shortener
- Build a Facebook chat
- Architect a worldwide video distribution system

Product Architecture or Design

During the product architecture or design interview, the interviewer will focus on building a product or API at scale that supports an end user product or service. You should be familiar with the areas below, but we're not looking for you to be an expert

in all of them. You should know enough to weight design considerations (and understand when to consult an expert) for:

- Storage data models
- Scalability
- Design patterns
- Data ownership
- Protocols
- Data formats
- Client-server design
- Designing for long-term vs. complexity
- Accommodating possible product changes

Example product architecture or design questions:

- Design a service or product API
- Design a chat service or a feed API
- Design an email server

Behavioral Interview

What can you expect?

The behavioral interview will consist of a 45-minute session. Your interviewer will want to learn about your background, what you're passionate about in tech, and what kind of impact you want to make.

What do we look for?

The purpose of the behavioral interview is to assess if a candidate will thrive in Meta's fast-paced and highly unstructured environment. To that end, we assess candidates on five signals that correlate with success at Meta:

- **Resolving conflict:** What kind of disagreements have you had with colleagues and/or managers? How have you resolved them? Can you empathize with people whose points of view differ radically from yours?
- **Growing continuously:** Your interviewer will be assessing your aptitude for seeking out opportunities for growth and learning. Do you take constructive criticism as an opportunity to improve? How have you approached improving your skills?
- **Embracing ambiguity:** How do you operate in an ambiguous and quickly changing environment? Are you comfortable making decisions and

maintaining high levels of productivity when you are missing information or lack clarity? How did you react when you had to quickly pivot away from a project due to a shift in priority?

- **Driving results:** We will be evaluating your experience pushing yourself and others to deliver against goals and objectives. How do you demonstrate your impact? Are you self-directed in reaching goals despite challenges and roadblocks?
- **Communicating effectively:** How well do you communicate with teams? What about cross-functional partners? How do you tailor your communications based on the work and/or the audience?

We may ask you to:

- Discuss available details of past and current projects
- Provide specific examples about what you did and the resulting impact.
- Share what you learned from a past situation.
- Talk about what you like about your current role and/or being a developer.

How to prep

Just like with the coding and design interviews, it's important to prepare ahead of time for interviews designed to get to know you better.

- **Know yourself.** Take the time to review your own résumé as your interviewer will almost certainly ask about key events in your work history. Be prepared to discuss projects in depth. It's helpful to outline 2-3 major projects ahead of time.
- **Be honest.** Not every project is a runaway success, and we may not always interact perfectly with our peers. Being transparent in these situations won't be counted against you in the interview. In fact, sharing & discussing how you learned, improved, and grew from your past experiences is valued. Share what you learned from a past situation.
- **Use the S.T.A.R. method** to mentally organize your thoughts. This will provoke a well-thought-out and chronological action of events. Easy to describe, easy to follow. You can practice this method using the Journaling Exercises in the Preparation Hub withing your Career Profile.
    - **S**—One or two sentences about the **SITUATION:** What happened?
    - **T**—Describe the **TASK:** What was your specific goal?
    - **A**—**ACTIONS** you took to overcome the obstacles and complete your objective.
    - **R**—The tangible / quantifiable **RESULTS** of the situation: How did it help

the team / company?

- **Have concrete examples or anecdotes.** Support each question with practical experiences and examples. Avoid theoretical answers - if you go into a theoretical tangent, your interviewer will redirect you to provide a concrete example.

Post Interview – What to Expect

You can expect your recruiter to provide a specific timeline or updates along the way. Your recruiter will inform you of next steps after your interview as soon as they are available. Feel free to follow up with them if you have not heard within a week of your interviews.

Appendix / Resources

Below you will find some helpful resources for your interviews. Take a look through the list as you prepare.

About Meta

- About Meta website
- Meta Life website
- Meta Diversity website
- Meta Quaterly Earnings
- Join our Meta Careers Talent Community page for the latest updates.
- Request to interact with an employee and learn what it's like to work at Meta through the Meta Connections Program
- Interviewing at Meta: The keys to success blog
- Meta Employee Benefits website
- Meta Newsroom website

Connect with Meta Employees

- Once you've made it to the onsite interview stage, request to interact with an employee and learn what it's like to work at Meta through the Meta Connections Program.

Meta's Technical Environment

- Engineering at Meta - FB page

- Meta Code videos
- Meta Opensource website
- Engineering Leadership at Meta blog

Coding Resources

- Software Engineering Interview Q+A Video
- Cracking the Meta Coding Interview Videos (The Approach and Problem Walk-through) (password: FB_IPS)
- Topcoder
- GeeksQuiz

Design Interview Resources

- System and Product Design Interview overview
- Grokking the Design InterviewTopcoder
- GitHub Design Primer
- Familiarize yourself with Excalidraw

Update personal information, track interview progress, and send thank you notes.

At any time during the interview process, you can track your progress, send thank-you notes and update your personal information all via the Career Profile. If you do not receive a link from recruiting, you may create one.

Thank you for taking the time to review this guide!