Application

Resume Review

- If you're applying for any of our positions - surprise! - real humans are reading your resume!
- We're not looking for buzzwords nor long lists of tools or technologies; there's no resume scanner here to pick up on them.
- We are looking for what you've worked on and how you accomplished it. What did you specifically design and implement? What was your team like? What did you learn and gain experience in? What obstacles did you face?
- We want to know what you want to work on next. If you're thinking about a specific role, make sure to include your relevant experience.
- If we don't think this position is a match for your experience, we'll try to figure out where else you might fit. If we don't have an opening right now, we'll let you know - but we recognize it's all about the right role at the right time and because we're growing quickly, we always expect for more roles to open up.

Introduction To Bina

- Feel free to reach out to us directly! Send us an e-mail: [link]. We don't bite and we're happy to chat with you or answer any questions have.
- It's likely the first member of Bina you talk to will be from the talent team. We're not looking for high-level discussions or stepping through your resume - we're trying to get to know you. Here are some things we think about:
    - What have you been working on most recently?
    - What are the technical obstacles you face - down to the gritty details? How did you navigate your way through that situation?
    - What is actually important to you in your next role? No, seriously - what do you want to do? What do you hope your team and the collaboration looks like? We're trying to figure out what team you'd be the happiest on and who we should introduce you to.

Technical Phone Interview

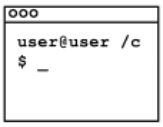
Past Experience

- Be prepared to talk about your design decisions - how your broke down and thought through problems you solved (big or small).

- Talk about the impact of your work as well as what was important to you about it. What was the most difficult part of what you worked on? What was your team like? What did you learn that has shaped you as an engineer?

Technical Discussion

- The below are topics that are generally covered - it doesn't include every topic and some may be excluded for your interview depending on the type of position, but this should give you a good starting point.
- General (all positions)
    - Data Structures (Array, LinkedList, HashMap, Trees, Stack/Queues, Heaps, Graph, etc)
    - Algorithms and analysis of complexity (Big-O analysis in terms of time and space)
    - Logic and problem solving skills: trade-offs and optimization
- Backend topics
    - Object-oriented design (e.g. inheritance vs composition, design patterns, etc)
    - Databases: relational vs non-relational; query optimization, schema design
    - Java: language, implementation, virtual machine, frameworks, run-time environment
- Frontend topics
    - Javascript, HTML/DOM Scripting, CSS
    - Single-page applications
    - UI components and interaction with the APIs



Coding

Solving The Problem

- Make sure to clarify any questions you have about the problem definition.
- Break down and outline your logic for your solution to your interviewer.
- Check your implementation along the way - test your code!
- If you're stuck, explain to your interviewer what parts you have solved but what other information you need to finish what you're stuck on. It's better to ask for help than stay stuck for too long.
- If you have extra time, you should be able to think about alternate ways to do the problem and talk about trade-offs. For example, if you had used an alternate data structure, algorithm, or other implementation what would be better or worse?
- Be able to take feedback and iterate that into your solution. For example, if your interviewer hints that there's a better solution and gives you something you didn't take into

account, this is a great time to discuss with them and come up with something together. That's something the engineering teams do together day-to-day!

Writing Code

- A great place to practice is LeetCode (https://leetcode.com) - they have a ton of free problems, let you practice in almost any language, and it gives you feedback for any solution that you submit.
- Writing clean code is really important to us. Make sure your code is easy to read, the logic flow makes sense, your naming conventions are appropriate, etc. We're thinking about when we have to do code reviews with you
- Pace yourself. Don't worry about micro-optimizations that make little difference - especially if you haven't solved the problem yet. For example, if you have a small check executing twice, it's probably not worth spending a long time re-doing that if you haven't gotten the majority of the problem. You can always go back and take care of those cases at the end.

On-site Interview

Coding Interviews

- Part 1: Your interviewer will define the problem with you and have you outline what your solution might be (likely in simplified pseudocode).
- Part 2: Implementation! We'll provide you with a laptop, set up the environment, and have what's on your screen projected. We often write code together this way as a team.

Design and Architecture Interviews

- Be prepared to whiteboard - drawing out your ideas is really important.
- Talk out your large-picture outline first, then go over details for each part.
- Note any assumptions you are making.
- Think about any drawbacks to your design and how it might be further optimized. Be prepared to talk about trade-offs.