I did the Palantir on-site a few weeks ago and it was pretty straightforward. The first round was a open-ended question on designing some application (i.e an elevator system, calendar app, job application portal, etc). I discussed tradeoffs, what features we should support, main use cases, possible edge cases, what API endpoints I would support, and how these endpoints would be glued together. No actual coding was necessary.

The second round was a debugging session, working with network calls/concurrency in Python. They gave you some initial code that was badly written and had bugs in it, and they had you find the bugs, explain your thinking process, and refractor it.

OA is string processing or graph question

1 round system design, no coding. 1 hour algorithms question (LC med/hard level). 1 hour learning interview, which was kinda unique. You read documentation for a concept they think you won't be familiar with based on your resume and solve a problem with the new knowledge.

OA -> Karat -> HR screen (please research the company before this, at least know the difference between a dev and a delta - although I would also recommend getting familiar with their three core products) -> 2 on-site technicals (won't give specifics due to NDA) -> 1 additional on-site if you fail one of the first 2 -> hiring manager call (technical).

Karat is a 3rd party interviewing platform so your interview will not be with a Palantir engineer. The interview consists of 5 mins for introductions, 10 mins for 3 time complexity questions, and 45 mins for as many Leetcode-style questions as you can solve. The interviewer was pretty nice; however, one thing to keep in mind is that they are very strict when it comes to the time limits. I solved 2 questions and gave my thoughts about the third but didn't have enough time to write code.

HR call:

Typical HR call with behavioral questions you would expect about your interests, resume, etc.

(Virtual) Onsite:

3 back-to-back 1hr interviews:

1. Decomp: Break down a high-level problem. No code is necessary. Was like a system design interview but more high-level.
2. Learning: The interviewer introduces a concept that they think you didn't encounter based on your resume and expects you to apply said concept to solve problems.
3. Coding: Typical Leetcode interview. Expect Med-hard.

Each of the aforementioned interviews began with a 10-15 min behavioral part.

Hiring manager:

Half coding and half behavioral. Scheduled for 1hr but took 15 mins longer. I think this interview is where I stumbled and was ultimately the reason for my rejection.

**Why are interviewers at Palantir obsessed with this: For an array of integers, give an algorithm to determine if there are 3 elements that sum to zero. What are the time and space complexity? Generalize to the case where the sum of k elements is 0?**

Given four arbitrary points (x,y) implement a function to determine if they form a square.

Build a Dependency Graph Version A:

Array of strings, every group of 4 strings is 1. “Task: *taskname*” 2. “Files: *file1 file2 file3…*” 3. “Dependencies: *taskA taskB taskC*…” 4. Empty String for human readability“”

Files are being changed, if file is changed, task with that file must be run. All dependencies must be run first before a task is run (some tasks have no dependencies)

Globs, two approaches 1. Using recursion 2. Using regular expression matching

For dependencies, used breadth-first search, first got the tasks to be run (their files were changed) with no dependencies, then the tasks to be run whose dependencies are either tasks with unchanged files, or tasks that have already been ran. Each successive layer, finally deepest layer is the one with tasks needing the deepest dependencies.

Absolute and relative file names