5 Kinds of Interview Rounds

Coding | LLD | HLD | Leadership | Resume

Sample Feedbacks

- **Feedback Details and Meeting Notes**:
  1. Problem Solving: DSA, Time complexity analysis = 4/5
     a. Was able to understand the problem quickly. and come up with test cases and figure out what would be answer in those cases
     b. Was able to come up with the brute solution with minimal hints.
     c. Understand time/space complexity of his solution.
     d. When asked for alternate solution. He came up with a greedy solution. He explained to me very briefly, and said he can code it It was working fine on sample example.
     e. He didn't have proof for his solution, neither any logic on why it should work.
     f. Solution looked incorrect to me, asked him to check on other cases.
     g. Need to provide the test cases, he was able to understand and rectify his mistake.
     h. After some hints, able to come up with the optimal solution.
  2. Implementation: Coding, Debugging and Code Quality = 3.5/5
     a. Candidate showed knowledge of java by writing code.
     b. Code quality - Do the code review and add your review comments here.
     c. Time to Implementation - Longer/ sufficient/ fast
     d. Had working Code -
     e. Able to understand his own code -
     f. Debugging skill - Inlign
  3. Tech Theory- (JAVA/ React/ JavaScript/TypeScript) = 4.5/5
     a. Provided satisfactory answers to -
        i. Error vs exception in Java
        ii. etc.
     b. Not Satisfied with response on -
        i. How to create immutable class?
        ii. etc.
  4. Miscellaneous: Introduction, Role Alignment, Communication, Explanation etc. = 4.5/5
     a. Candidate joined on or before time. or candidate joined X minutes late.
     b. From Introduction, their profile looks very much aligned with the role. Not worked much on React but have basic knowledge.
     c. Able to explain solution - clearly / struggled.
     d. Confidence/ Attitude / Receptive of Feedback or hints/ Good Listener / Respectful.
     e. Values - Professional Integrity, Ethical Standards, Honesty / Taking help from internet /AI/ or someone.
     f. Any other thing that couldn't be fit in technical sections.

- **LLD Round:**
  - **Summary**: Inclined
  - **Code report**: Link to report
  - **Feedback Details and Meeting Notes**:
    - **LLD: Requirement Gathering, Conceptual Understanding, Object-Oriented Design (Classes, Design Patterns, Inheritance, Composition, Extensibility, Code, Testing) = 10/15**

      Requirement Gathering:
      The candidate was able to quickly understand the problem statement and ask relevant clarifying questions.
      He demonstrated a good understanding of functional requirements but missed a few edge cases related to non-functional requirements (like scalability and performance).

      Design Approach:
      Presented a well-structured design using standard Object-Oriented principles (OOP).
      Applied **Inheritance** and **Composition** effectively where appropriate, and demonstrated awareness of code extensibility.
      Discussed the use of **Design Patterns**, opting for a **Factory Pattern** to solve a problem but required hints to justify why it was necessary.

      Extensibility and Code Design:
      Initial design was not fully extensible to accommodate future changes or requirements. After some feedback and hints, the candidate revised the design to make it more modular.
      Demonstrated a sound understanding of **SOLID principles** but could improve on applying these principles more consistently throughout the design.

      Code Quality:
      Coding conventions were followed, and the candidate showed proficiency in writing clean, maintainable code.
      Class interactions were mostly well thought out but could have been better tested for corner cases.

      Testing and Validation:
      The candidate was able to identify and suggest appropriate test cases for his design, both positive and negative. However, some cases involving boundary conditions were not immediately considered until prompted.
    - **Theory** -
      - **Strengths:**
        - Provided clear explanations for common OOP concepts such as **Inheritance**, **Polymorphism**, and **Abstraction**.
        - Gave a satisfactory answer on the **Decorator Pattern** and **Singleton Pattern** and where they should be applied.
      - **Weaknesses:**
        - Could not provide a solid example for **Dependency Injection** and its benefits when asked about scalable architecture design.
        - Was somewhat unclear about the difference between **Aggregation** and **Composition**.

- **HLD Round:**
  - System Design Skill:

    Overall, the candidate demonstrated strong proficiency in high-level system design.

    Details:

    - Requirement Gathering:
      - Listened to the problem statement, then asked clarifying questions before proposing solutions.
      - Clearly captured functional and non-functional requirements.
      - Stated reasonable assumptions to frame the design.
    - Architecture Diagram:
      - Produced a coherent high-level architecture including core components (e.g., load balancer, API gateway, services, queues, databases, cache, object storage, pub/sub, CDN).
      - Initially missed a few components but incorporated them promptly after feedback.
      - Was able to effetely draw the high-level design.
    - Depth, Detailing & Discussion:
      - Provided a clear, detailed walkthrough of the high-level design and component responsibilities.
      - Completed capacity estimates (e.g., storage and compute needs etc) with sensible reasoning.
      - Discussed trade-offs of the chosen approach and proposed viable alternatives.
      - Explained component interactions accurately and confidently.
      - Was able to Deep dive of one major component from the proposed design.
    - Theory:
      - Demonstrated solid understanding of core concepts such as load balancing, CAP theorem, sharding, scalability, availability, and communication protocols etc.

3.1. Technical (~30 mins) :

3.1.1. Project Architecture: 0

1. Candidate was able to draw the architecture, but it have various missing components, involving DB, Cache, KAFKA producer consumer were not clear. However on cross questioning he was able to explain and correct it.
2. Candidate is familiar with current architecture, but not much involved in system design; lacks clarity on why another alternative is not used. Why they need KAFKA?
3. From the architecture perspective, candidate's team work doesn't look like more technically challenging. Owning a service, which is more or less dependant on other services to do the major work.
4. Asked questions regarding, offloading reducing 3rd party spend which they are using as a rule engine, candidate mentioned they have planned and estimated 2 years of efforts.
5. Probed into multiple architectural alternatives, and candidate lacked reasoning to justify why or why not? It seems like they have gaps and whenever I proposed an alternative solution, candidate was more into it was build this way, and aligned on it is better but can't do anything.

3.1.2. System Design Components : 0

- Asked about Redis and DynamoDB
  - Basic understanding of it.
- KAFKA, SQS, RabitMQ
  - Lacks clear understanding of use cases where each would be good fit.
  - Familiar about this alternatives, but not sure on any details.
  - No Knowledge about internal details.

3.2. Behavioral (~30 mins): (~5 mins per values)

3.2.1. Mentorship : -1

- Talked about current team structure, his role and mentorship involvement.
- On asking about challenges he faced while mentoring - used strong negative word for the junior he mentored as a lead you should be presenting in the right way and defend your team. Mentioned they were campus hire, Not industry ready or industry idea etc.
- To work with them had weekly connect and pair program rather than creating process documentations.
- Most of things should have been solved by defining a clear process and documentations upfront, which candidate lacked. Although he have good amount of mentorship experience.

3.2.2. Conflict : -0.25

- mentioned conflicts between SRE and dev team regarding requirement to change in dynamoDB write those changes to KAFKA Topic.
- Asked for any other example where he with his seniors or managers ; mentioned NO conflicting situations.

3.2.3. Critical Feedback: 0

- Do not handhold and let them work while mentoring.
- Requirement to be clear before starting, and additional requests from them and leads to the issue from tester side.
- Communication Gap with the product manager / owners. Documentations related.
- Learned and document discussion, using channel tech leads, managers and stackholders, being proactive, not working independently and with the team.

3.2.4. leanings from career : +0.5

- Software is ever evolving and can't be comfortable and communications
- team game >> individual game.
- ownership and take responsibility.

3.2.5. Take Risk : +1

- Updation by us every time, proposed UI application solution to simplify and automate.
- Risk was even though he proposed, he wasn't familiar with UI..
- Did the development and pitched, the product went live.

Debrief

Initial Feedback is submitted.
Voting - Yes, No, Slow Down.

Round of discussions

Final Decision
* Hire / No Hire / Up-level (rare) / downlevel

Things that feels like Illegal to Know

1. Everyone is Busy !! HM is the most busiest for a position
    10 profiles are shortlisted.
    first couple are just for seeing the market.

2. Scheduling all rounds in a Day.
    * You are busy too, and work focussed. Taking just a day off for the Interview.
    * Not Much Like you need to over prepared.
    * In terms of conflicting feedback - it will support you.

3. Asking for Offline Interviews :
    * Remove suspecious.
* cheating doubts is now discussed and called out in
    40% + interviews, compared to previously 5-10% candidates.

4. Who owns the room?
    * HM and Bar raiser decides the outcome.
    * Debrief decision --
    * role of bar raiser is to ensure all feedback are equally important.
     * but often a leader / sr. eng being more vocal and having more experience/ respect
        - influence more.

5. Prior Research and asking Good questions
    * Leadership principles
    * Listen to their intro and ask questions accordingly.

6. Writing keywords as you say
    * your cursor remains in the screen.
    * doubt of cheating.
    * easy for interviewers to take notes. Avoid misunderstanding in critical cases - time time complexity.
    * understand that interview has to fill out the positives and gaps areas.

Major Red Flags That Changed the Decisions

1. Luck
2. Not able to provide enough data points.
3. Communications
4. Incorrect Basic Things
    * Time complexity wrong
5. Integrity and Trust - Use of external source / AI.