**SDE Profile**

**Data Structures & Algorithms (DSA)**

**Interview Questions**

* Implement a map (hashmap from scratch).
* Reverse an array in least time complexity.
* Factorial using recursion.
* Linked list implementation (create, insert, print nodes).
* Find all triplets with sum = 0 (3-Sum Problem).
* Analyze time complexity of given code.
* Implement a key–value store with timestamp-based tracking.
* Implement LRU Cache.
* Data structure behind a Merkle Tree (→ Hash Tree).

**Related Questions**

* Binary search (recursive & iterative).
* Detect cycle in a linked list (Floyd’s cycle detection).
* Compare Heap vs Binary Search Tree (BST) – use cases.
* Implement stack/queue using arrays or linked lists.
* Difference between BFS and DFS (with coding examples).
* Design a data structure for autocomplete/suggestions (Trie).

**SQL / Database Questions**

**Interview Questions**

* PL/SQL query to check if a string is palindrome.
* Query to print duplicate rows.
* Resume-based SQL/DBMS questions.
* SQL vs NoSQL.
* General DBMS fundamentals.

**Related Questions**

* SQL query for second highest salary (with and without LIMIT).
* Normalization: 1NF, 2NF, 3NF (with examples).
* Clustered vs Non-clustered indexes.
* ACID properties of transactions.
* CAP theorem (in context of NoSQL).
* Difference between DELETE, TRUNCATE, DROP.

**Python / Programming Concepts**

**Interview Questions**

* Commands & implementation details of Python projects.
* Python code for NLP/Speech project (resume-related).
* Recursion debugging in Python.

**Related Questions**

* Shallow copy vs Deep copy in Python.
* Python decorators (use cases in logging, authentication).
* Python’s GIL (Global Interpreter Lock) – why it exists.
* List comprehension vs generator expression.
* Exception handling (try-except-finally).
* Multithreading vs Multiprocessing in Python.

**OOPS & CS Fundamentals**

**Interview Questions**

* Overloading vs Overriding.
* Abstraction (real-life examples).
* General OOPS concepts.
* CS fundamentals (DBMS, OS, Networking basics).

**Related Questions**

* Encapsulation (with example).
* Interface vs Abstract class.
* Polymorphism in real-world applications.
* Process vs Thread.
* Deadlock (necessary conditions).
* Paging vs Segmentation (OS).

**Project-Based Questions**

**Deep-dive into your resume projects**

* Explain your project: modules, functions, code structure.
* Full-stack: JWT, React, JS (auth flow).
* GitHub/Linux commands used in your project.
* Resume-based technical deep dive.
* “How would you scale your project if users increased 100x?”
* “How would you secure API endpoints?”
* “If one module fails, how will the system handle it?”

**HR / Behavioral Questions**

**Interview Questions**

* Tell me about yourself.
* Why our company?
* What do you expect to learn in this role?
* Latest technologies/trends in market.

**Related Questions**

* Why do you want to join as an SDE?
* Example of solving a technical challenge under time pressure.
* Collaborative project experience (Git, handling merge conflicts).
* Biggest mistake in a project & how you corrected it.
* Strengths & weaknesses.

**Preparation Strategy**

1. **DSA:**
   * Practice LeetCode medium-level questions (maps, arrays, linked lists, caching).
   * Focus on time complexity + optimization approaches.
2. **SQL/DBMS:**
   * Revise joins, subqueries, indexing, transactions.
   * Practice queries on duplicates, nth highest salary, group by.
3. **Python/OOPS:**
   * Revise decorators, iterators, memory management.
   * Practice class-based design problems.
4. **Projects:**
   * Be able to **explain end-to-end** (problem → solution → tech stack → impact).
   * Keep 1-2 backup projects ready.
5. **HR:**
   * Prepare STAR (Situation, Task, Action, Result) stories for behavioral questions.
   * Research Amex culture, tech initiatives (AI/ML, FinTech, cloud).