ROADMAP LLD (Live Cohort below)

Phase 1: OOP Foundations (Week 1)

Must-know concepts

- **Encapsulation (private state, public behavior)**
- **Abstraction vs Interface**
- **Inheritance (when NOT to use it)**
- **Polymorphism (runtime > compile-time)**
- **UML Diagrams**

Interview expectations

- **Correct class responsibility**
- **Proper access modifiers**
- **Clean constructors**

Practice mini-designs

- **Design a Printer**
- **Design a BankAccount**
- **Design a Car with Engine abstraction**

Phase 2: SOLID Principles (Week 2)

This is non-negotiable for SDE-2.

SRP – Single Responsibility

One class = one reason to change

OCP – Open Closed

Open for extension, closed for modification

LSP – Liskov Substitution

Child should be replaceable by parent

ISP – Interface Segregation

Many small interfaces > one fat interface

DIP – Dependency Inversion

Depend on abstractions, not concretes

Always explain what problem each principle solves.

Phase 3: Design Patterns (Week 3–4)

Focus on when & why, not memorization.

Creational

- **Singleton (thread-safe)**
- **Factory**
- **Abstract Factory**
- **Builder**

Structural

- **Adapter**
- **Decorator**
- **Facade**
- **Composite**

Behavioral (Very important)

- **`Strategy`** ⭐⭐⭐
- **`Observer`** ⭐⭐⭐
- **Command**
- **State**
- **Chain of Responsibility**

Most LLD problems reduce to Strategy + Factory + Composition

Phase 4: Concurrency in LLD (Week 5)

SDE-2 often includes thread-safety discussion.

Must-know

- **synchronized**
- **volatile**
- **Lock**, **ReentrantLock**
- **Executors**
- **Thread-safe Singleton**
- **Producer-Consumer**

Apply in designs

- **Thread-safe Cache**
- **Rate Limiter**
- **Parking Lot entry system**

Phase 5: Core LLD Interview Problems (Week 6–8)

These are absolute must-designs

Beginner

- **Design a Logger**
- **Design a File System**
- **Design a Vending Machine**

Intermediate (SDE-2 Sweet Spot)

1. **Parking Lot System**

2. **Elevator (Lift) System**
3. **LRU Cache**
4. **Rate Limiter**
5. **Online Chess Game**
6. **Splitwise (Expense Sharing System)**
7. **Movie Ticket Booking System**
8. **ATM System**
9. **Notification System**
10. **File System (Simplified)**
11. **Ride Booking System (Uber/Ola)**
12. **URL Shortener**
13. **Logger System**
14. **Vending Machine**
15. **Online Shopping Cart**
16. **Hotel Booking System**
17. **Snake and Ladder Game**
18. **Library Management System**
19. **Task Scheduler**
20. **Cache with Expiry (TTL Cache)**

Enroll:https://forms.gle/3tWy6qURHeyRTVmG9

System Design Roadmap[These are must rest is optional]

Foundations

- Client–Server model
- HTTP / REST basics
- Latency vs Throughput
- CAP theorem

Scalability Basics

- Vertical vs Horizontal scaling
- Stateless vs Stateful services
- Load balancing basics

Databases

- SQL vs NoSQL
- Indexing & sharding
- Replication & consistency
- Read vs write optimization

Caching

- Why caching is needed
- Cache-aside pattern
- Redis / Memcached basics
- Cache invalidation strategies

API Design

- RESTful API design

- Idempotency
- Pagination & filtering
- Versioning

Asynchronous Processing

- Message queues
- Pub–Sub model
- Event-driven architecture
- Background jobs

Consistency & Reliability

- Strong vs eventual consistency
- Idempotent consumers
- Retry, timeout, circuit breaker

Distributed Systems Concepts

- Leader election
- Distributed locking
- Heartbeats & failure detection

Security & Observability

- Authentication & authorization
- Rate limiting
- Logging, monitoring, alerting

Interview Questions

- URL Shortener

- Chat application
- Feed system
- Rate Limiter
- File storage system

15 HIGH-SIGNAL LLD INTERVIEW PROBLEMS (NON-CLICHÉ)

🔁 CONCURRENCY / MULTI-THREADING (VERY HOT 🔥)

1️⃣ Rate Limiter (Token Bucket / Sliding Window)

📌 Asked at: Google, Uber, Atlassian, Amazon
Tests:

- **Thread safety**
- **Time-based logic**
- **Atomic operations**
- **Strategy Pattern**

Variants: per-user, per-API, distributed

2️⃣ Thread-Safe In-Memory Cache (LRU + TTL)

📌 Asked at: Amazon, Flipkart, Microsoft
Tests:

- **ConcurrentHashMap**
- **Locks vs lock-free**

- **Eviction policies**
- **Background cleanup thread**

3️⃣ Producer–Consumer Queue with Backpressure

📌 Asked at: Amazon, Netflix
Tests:

- **Condition variables**
- **Blocking queues**
- **Bounded buffers**
- **Fairness**

4️⃣ Scheduled Task Executor (Mini Cron)

📌 Asked at: Google, Uber
Tests:

- **DelayQueue**
- **Thread pools**
- **Priority scheduling**
- **Time abstraction**

5️⃣ Pub-Sub System (In-Memory)

📌 **Asked at: LinkedIn, Uber**
Tests:

- **Observer pattern**
- **Thread-safe fan-out**
- **Message ordering**
- **Async delivery**

6️⃣ Thread-Safe Connection Pool

📌 **Asked at: Oracle, Amazon**
Tests:

- **Object pooling**
- **Semaphore**
- **Timeout handling**
- **Resource cleanup**

🧠 STATE + CONCURRENCY (INTERESTING COMBOS)

7️⃣ Multiplayer Game Lobby System

📌 **Asked at: Microsoft Gaming, Amazon**
Tests:

- **Concurrent joins**
- **Matchmaking**
- **State transitions**
- **Lock granularity**

8️⃣ Online Voting / Polling System

📌 **Asked at: Meta, Atlassian**
Tests:

- **Atomic updates**
- **Idempotency**
- **Race conditions**
- **Consistency guarantees**

9️⃣ Ride-Matching Engine (Core Logic Only)

📌 **Asked at: Uber, Ola**
Tests:

- **Concurrent updates**
- **Strategy pattern**
- **State machines**
- **Fair matching**

🔔 EVENT / ASYNC SYSTEMS

🔟 Notification Dispatcher (Email / SMS / Push)

📌 **Asked at: Amazon, Google**
Tests:

- **Retry logic**
- **Async queues**
- **Thread pools**

- **Failure handling**

1️⃣1️⃣ Distributed ID Generator (Snowflake-like)

📌 **Asked at: Twitter, LinkedIn**
Tests:

- **Thread safety**
- **Time ordering**
- **Bit manipulation**
- **Clock drift handling**

1️⃣2️⃣ Circuit Breaker

📌 **Asked at: Netflix, Amazon**
Tests:

- **State pattern**
- **Concurrent counters**
- **Timeout windows**
- **Failure thresholds**

🎯 DESIGN + REAL-WORLD CONCURRENCY

1️⃣3️⃣ Log Aggregator (In-Memory)

📌 **Asked at: Splunk-like companies**
Tests:

- **Concurrent writes**

- **Buffering**
- **Batch flush**
- **Memory limits**

1️⃣4️⃣ Job Scheduler with Dependencies

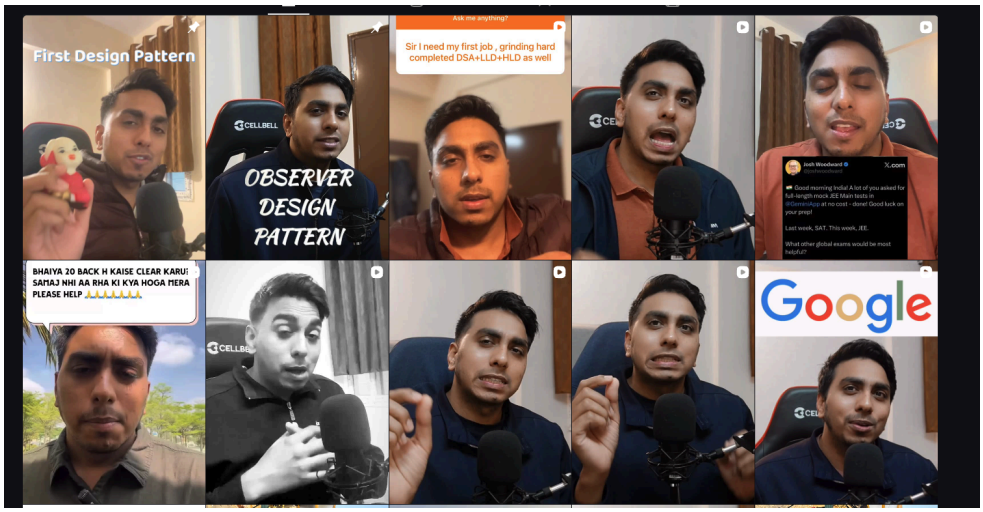
📌 Asked at: Airbnb, Google
Tests:

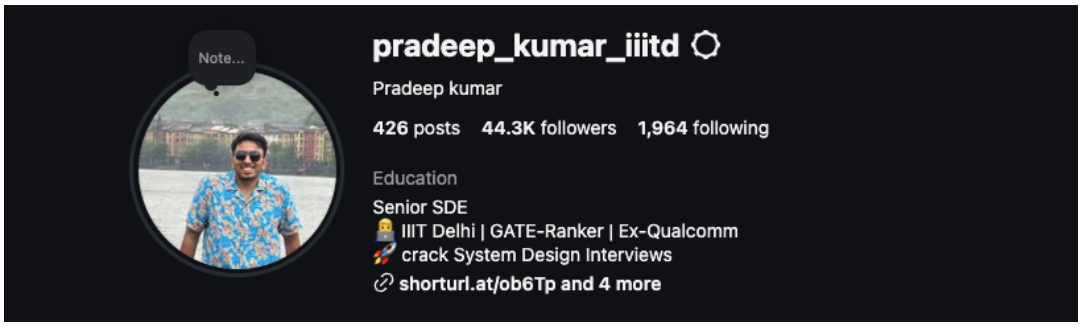
- **DAG handling**
- **Thread pools**
- **Deadlock prevention**
- **Dependency resolution**

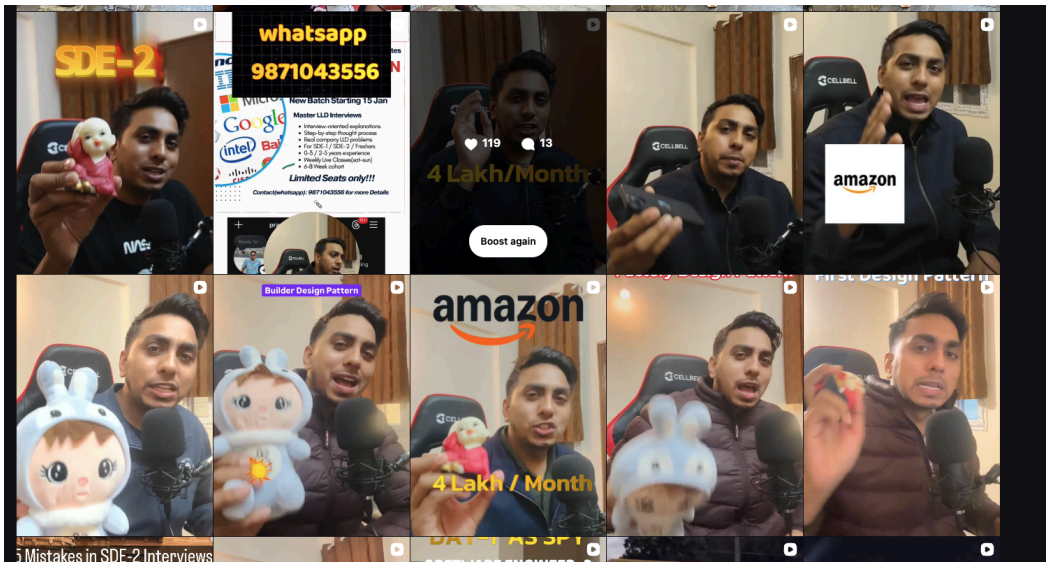
1️⃣5️⃣ Chat System (Core Engine Only)

📌 Asked at: Meta, Amazon
Tests:

- **Message ordering**
- **Concurrent delivery**
- **Fan-out models**
- **Consistency tradeoffs**


First Design Pattern
OBSERVER DESIGN PATTERN
Sir I need my first job , grinding hard completed DSA+LLD+HLD as well
BHAIYA 20 BACK H KAISE CLEAR KARU SAMAJ NHI AA RHA KI KYA HOGA MERA PLEASE HELP
Google


Note...
pradeep_kumar_iiitd
Pradeep kumar
426 posts 44.3K followers 1,964 following
Education
Senior SDE
IIIT Delhi | GATE-Ranker | Ex-Qualcomm
crack System Design Interviews
shorturl.at/ob6Tp and 4 more


SDE-2
whatsapp 9871043556
Limited Seats only!!!
119 13
4 Lakh/Month
Boost again
amazon
Builder Design Pattern
amazon
4 Lakh / Month
5 Mistakes in SDE-2 Interviews