🚀 LECTURE 01: INTRODUCTION TO SYSTEM DESIGN - COMPLETE NOTES

📌 Main Points

- DSA padh li hai — ab **real-world applications** kaise banti hain, samajhne ka time aa gaya hai.
- DSA ki knowledge sirf **LeetCode problems** tak limited nahi honi chahiye.
- Real-world apps jaise **Swiggy, Zomato, Ola, Uber** kaise kaam karti hain?
- Backend ka structure kya hota hai?
- Millions of users ko ek saath **handle kaise karte hain?**
- FAANG/startups ke liye **LLD (Low Level Design)** ki knowledge **must** hai.

🔶 WHAT IS LLD? (Basic Definition)

📘 DSA vs LLD Analogy

👉 DSA isolated problems solve karta hai — jaise *Searching (Binary Search)*, *Sorting (Merge Sort)*.
👉 LLD unhi DSA concepts ko combine karke **poori application banata hai** — jaise *Complete Ride-Booking System*.

🧩 **In Simple Words:**
👉 "**DSA ke concepts ko milkar POORI APPLICATION banana = LLD**"

📖 STORY: ANURAG vs MAURYA

👥 **Scenario:** Company – *QuickRide* (Ola/Uber like platform)

Character	Skillset
👨‍💻 **Anurag**	DSA aati hai, LLD nahi
👩‍🎓 **Maurya**	DSA + LLD dono master

🔷 Anurag's Approach (Only DSA Perspective)

🧭 **Problem 1: Shortest Path**

- City = Graph
- Intersections = Nodes
- Roads = Edges

- Algorithm → **Dijkstra's**
  ✅ Problem solved

🚗 **Problem 2: Closest Rider Assignment**

- Riders → **Min-Heap (Priority Queue)**
- Distance calculate karke insert karo
- Heap se pop → closest rider mil gaya
  ✅ Problem solved

❌ **Manager's Feedback:**

> "Algorithms theek hain, par **Application kahan hai?**"
> "Entities, relationships, data flow, notifications, payments, scalability sab missing!"

⚠️ **Mistake:** Sirf algorithm socha, **application structure nahi.**

🔶 Maurya's Approach (DSA + LLD Combined)

✅ **Step 1: Identify Objects**

👤 **User –** Ride book karega
🚗 **Rider –** Ride provide karega
📍 **Location –** Coordinates track karega
🔔 **Notification –** Alerts bhejega
💳 **Payment –** Payment handle karega

✅ **Step 2: Define Relationships**

- User ↔ Rider (Connection)
- Rider ↔ Location (Tracking)
- Notification → Users & Riders (Messages)

✅ **Step 3: Think Real-World Scenarios**

- 🔐 **Data Security:** Personal numbers hidden
- ⚙️ **Scalability:** Handle millions of users
- 💬 **Integration:** Notification & Payment Gateway

✅ **Step 4: Use DSA Smartly**

- Structure ke baad algorithm apply karo

🎯 **Key Learning:**

> "**Pehle Blueprint banao, phir Tools use karo!**"

🏗️ LLD KE 3 MAIN PILLARS

1️⃣ 🌐 SCALABILITY – Badhna Aasani Se

📖 **Meaning:** System badhte users ke sath smoothly chale

Features:

- Millions of users handle kar sake
- Easily expand ho
- New features add karna easy ho

💡 **Example:**
1,000 → 1,00,000 users without crash.

2️⃣ 🛠️ MAINTAINABILITY – Sambhal Mein Aasani

📖 **Meaning:** Code aisa likho jo **easily maintain & debug** ho

Features:

- Naya feature add karo → purane na toote
- Easy debugging
- Modular aur clean code

⚠️ **Example:**
"Naya feature add kiya aur 4 purane toot gaye" ❌

3️⃣ ♻️ REUSABILITY – Dobara Use Kar Sakte Hain

📖 **Meaning:** Code reusable ho aur tightly coupled na ho

Concept: Plug & Play design

Examples:

- Rider Matching Algorithm → Swiggy, Zomato, Amazon Delivery
- Notification Module → Any app me plug ho sakta hai

🚀 **Pro Tip:**

> "Code likho jo kal kisi doosri app me copy-paste ho jaye!"

⚖️ LLD vs HLD - COMPLETE COMPARISON

Feature	LLD (Low Level Design)	HLD (High Level Design)
🎯 **Focus**	Internal code structure	Overall system architecture
🧩 **Questions**	Classes, Objects, Relationships	Tech stack, Database, Servers
⚙️ **Output**	Class diagrams, detailed logic	Architecture diagrams
🧠 **Level**	Code-level	System-level
💬 **Example (QuickRide)**	User class design	PostgreSQL or MongoDB? AWS or Azure?

💡 **Note:**

> HLD interview → mostly architecture discussion, **no coding.**

🔄 LLD vs HLD vs DSA – FINAL RELATIONSHIP

🧠 **DSA:** Brain – problem solving

🦴 **LLD:** Skeleton – blueprint & structure

💪 **HLD:** Body – overall architecture✅ **Together:** Complete application build hoti hai

❌ **Difference:**

- HLD → System architecture
- LLD → Code structure
- DSA → Logic & algorithms

🎖️ **Golden Line:**

> "**DSA is the Brain of Application, LLD is the Skeleton.**"

📚 POWERFUL SUMMARY

🎯 LLD Roadmap

```
REQUIREMENTS → OBJECTS → RELATIONSHIPS → SECURITY → SCALABILITY → DSA
→ TESTING
```

🔑 Key Takeaways

✅ LLD Kya Hai?

- DSA concepts ko combine karke **real-world applications design** karna

✅ 3 Pillars:

- 🌐 Scalability → Grow easily
- 🛠️ Maintainability → Maintain easily
- ♻️ Reusability → Reuse anywhere

✅ LLD vs HLD:

- LLD → Classes, Objects, Code structure
- HLD → Database, Server, System architecture

✅ Analogy:

- DSA = 🧠 Brain
- LLD = 🦴 Skeleton
- HLD = 💪 Body
