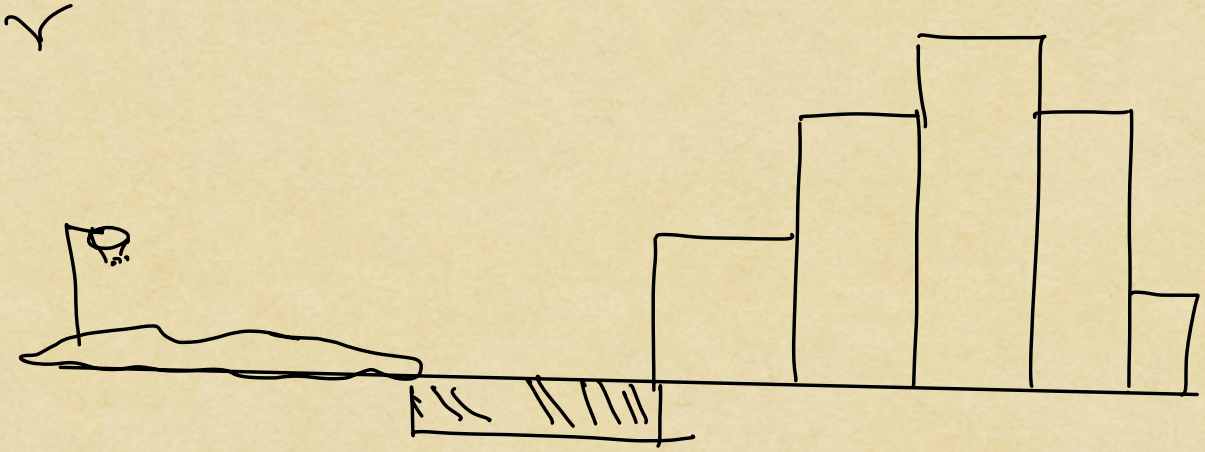
DESIGN =>

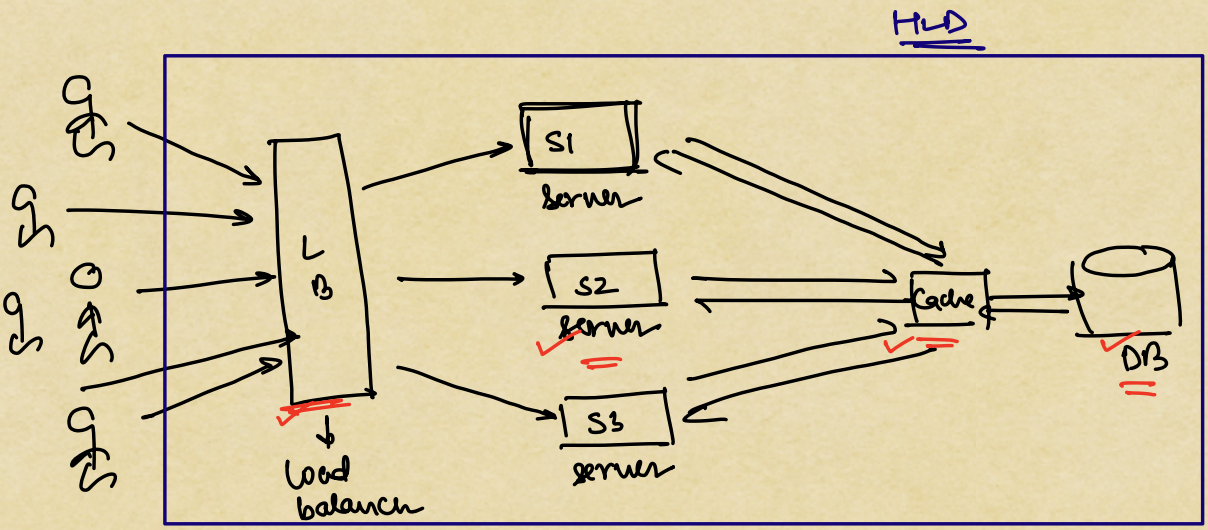
HLD
↓
High Level
Design

LLD
↓
Low Level
Design



HLD → Society

LLD → flat, swimming pool, garden
↓
design this individually



LB ✓ Server Cache ✓ DB

* all of these components are nothing much computers running some code.

* 6 same laptops can be converted into :-

  - 1 LB
  - 3 servers
  - 1 cache
  - 1 DB

  → completely doable.

* LLD we design the code/software that run of basic computers/machines and gives the capability to do certain actions.

* Why is LLD important :—

	≤5 YOE	>5 YOE
DSA	5	5
LLD (code + development) → (Interview + job)	85	75
HLD	10	20

100%.

* S/w in product company :-

→ development → code

→ bug fix → code reading, understanding, minor coding

→ testing → code reading, understanding

→ Pocs → exploration

→ emails → planning, estimations, product requirement

→ meetings → planning, estimations, product requirement

→ code review → code reading, understanding

→ documentations → code reading, understanding

→ lunch / dinner / tea / coffee] best part

→ gossipping

⇒ 100%.

↓

coding 12%

* coding
* reading code
* understanding code
* product requirement gathering
* planning and estimating
* designing

↓

LLD

PRD → product requirement doc

=> LLD in Interviews

Service based, Old banks, Old tech	FAANG, big tech companies	New tech startups
WITCH, Societe General, Barclays, HSBC, SBI	FAANG, Adobe, Oracle, Walmart, Visa, Atlassian	Flipkart, Swiggy, Cred, Scaler, Unacademy etc, Intuit
* theoretical * questions on syntax, keywords * minimum coding questions * questions on scenarios which can be solved by keywords	* very short problem statement to design * ask questions and do product requirement gathering * design this end to end * class diagram * schema design * pseudo code { 60 mins → 45-50 mins }	* very detailed problem statement (PRD) * design and build the project E2E with all features in PRD working with proper tests. * 120 mins → (100-110 mins) (machine coding round)

=> LLD module

→ LLD 1

LLD -1 (11)

- OOP(4 classes)
  - Concurrency & Multithreading
  - Advanced Language concepts
    - Java Streams
    - Lamda Functions
    - Collections

→ LLD-2

LLD-2 (10)

- Solid design principles(2 classes)
- Design patterns
  - Creational(2 Classes-5 design patterns)
  - Structural(1 Class- 3 design patterns)
  - Behavioural(1 class- 2 design patterns)

  We don't cover all design patterns, but the most important and Frequently asked ones are discussed.
- UML diagram(1 class)

10 DPs → 90%.

→ LLD-3

(15)

- Interview Prep & Practice
  - How to approach an LLD problem and Design a Pen
  - Design Tic tac toe & Code Tit tac toe
  - Design Parking lot & Code Parking lot
  - Design BookMyShow & Code BookMyShow
  - Design Splitwise

→ short problem Statement

↓

→ PRD → design → Code structure → do the coding

LLD 4

(20-25)

- Project
  - E-commerce Websites
    - Deploy
    - API's
    - No SQL DB
    - Security
    - Authentication

→ 4 microservices

→ SQL

→ Aws

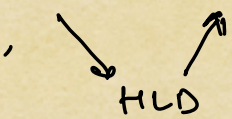
→ Redis / Kafka

→ Payment gtw

→ Authorisation (OAuth)

LLD 1 → 2 → 3 ⟶ 4

Or,



* Java + SpringBoot → Live
* Python + Django → (recordings) + TA
* ~~C# .Net~~
* NodeJs → Full Stack (MERN)

* Content + MOCK interviews ⇒ LLD 1 & 2
  LLD 3
  LLD 4

* assignments

* live coding sessions ⇒ follow through

* fundamentals → LLD
↓
OS | CN | DB

* SDE1 | SDE2