SYSTEM DESIGN TOPICS

▼ HIGH LEVEL DESIGN (HLD)

Step 1: System Design Fundamentals	DONE	Revisit
What is System Design?		
Serverless vs Serverful architecture		
Horizontal scaling vs Vertical scaling		
What are processes?		
What are threads?		
What are pages?		
How does the internet work?		
Client–Server architecture		
Stateless vs Stateful systems		

Step 2: Databases (HLD)	DONE	Revisit
SQL vs NoSQL databases		
Relational databases concepts		
NoSQL database types		
In-memory databases (Redis, Memcached		
Data replication		
Data migration strategies		
Data partitioning		
Sharding techniques		
Read replicas		

Step 3: Consistency, Availability & Reliability	DONE	Revisit
What is data consistency?		
Strong consistency		
Eventual consistency		
Isolation levels		
CAP theorem		
CAP trade-offs		
Real-world CAP examples		

Step 4: Caching Systems	DONE	Revisit
What is caching?		
Why caching is needed?		
Cache types		
Client side cache and server side cache		
Redis		
Memcached		
Write strategies		
Write through , Write-back , write around		

Cache eviction policies
LRU,LFU,FIFO,Segmented LRU
Content Delivery Networks (CDN)

Step 5: Networking Concepts

DONE	**Revisit**

TCP vs UDP
HTTP protocol
HTTP/1.1 vs HTTP/2 vs HTTP/3
HTTPS
TLS / SSL
REST communication
WebSockets
WebRTC
Video & live streaming basics

Step 6: Load Balancers

DONE	**Revisit**

Why load balancing?
Stateless vs Stateful load balancing
Load balancing algorithms
Round Robin,Least Connection,IP Hashing
Layer 4 vs Layer 7 load balancing
Consistent hashing
Proxy
Reverse proxy
Rate limiting
Throttling

Step 7: Message Queues & Asynchronous Processing

DONE	**Revisit**

Synchronous vs Asynchronous processing
Message queues
Kafka
RabbitMQ
Publisher–Subscriber model
Event-driven architecture
Exactly-once vs At-least-once delivery
Dead letter queues

Step 8: Monoliths vs Microservices

DONE	**Revisit**

Monolithic architecture
Microservices architecture
Why microservices?
Single Point of Failure (SPOF)
Cascading failures
Service discovery
Inter-service communication
Containerization (Docker)
Migrating monolith to microservices

Step 9: Monitoring & Logging | **DONE** | **Revisit**

Why monitoring is needed?
Logs vs Metrics
Centralized logging
Distributed tracing
Monitoring tools
prometheus, Grafana
Alerting systems
Anomaly detection
Health checks

Step 10: Security in System Design | **DONE** | **Revisit**

Authentication vs Authorization
Tokens for authentication
JWT
OAuth 2.0
Single Sign-On (SSO)
Access Control Lists (ACL)
Role-based access control (RBAC)
Encryption (at rest & in transit)
Secrets management

Step 11: System Design Trade-offs | **DONE** | **Revisit**

Push vs Pull architecture
Consistency vs Availability
SQL vs NoSQL
Memory vs Latency
Throughput vs Latency
Accuracy vs Latency
Cost vs Performance
Scalability vs Simplicity

Step 12: High-Level Design Practice (HLD) | **DONE** | **Revisit**

Design end-to-end systems for:
YouTube
Twitter
WhatsApp
Uber
Amazon
Dropbox / Google Drive
Netflix
Instagram
Zoom
Booking.com / Airbnb

▼ LOW LEVEL DESIGN (LLD)

Step 13: Object-Oriented Programming (LLD)	**DONE**	**Revisit**
Encapsulation		
Abstraction		
Inheritance		
Polymorphism		
SOLID principles		
Real-world OOP modeling		

Step 14: Design Patterns	**DONE**	**Revisit**
Creational patterns		
Singleton		
Factory		
Builder		
Structural patterns		
Proxy		
Adapter		
Bridge		
Behavioral patterns		
Strategy		
Observer		
Command		

Step 15: Concurrency & Thread Safety	**DONE**	**Revisit**
Multi-threading concepts		
Thread-safe injection		
Locks & mutex		
Synchronization		
Producer–Consumer problem		
Race conditions		
Deadlocks		
Concurrent data structures		

Step 16: UML Diagrams	**DONE**	**Revisit**
Class diagram		
Sequence diagram		
Use case diagram		
Activity diagram		
Component diagram		
When to use which diagram		

Step 17: API Design (LLD)	**DONE**	**Revisit**
REST API design principles		
Request/Response modeling		

API versioning
Extensibility
Idempotency
Pagination
Error handling
Clean code principles
DRY
SRP
Avoiding God classes

Step 18: Common LLD Interview Problems	**DONE**	**Revisit**
Design Tic-Tac-Toe / Chess game		
Design Splitwise		
Design Parking Lot		
Design Elevator System		
Design Notification System		
Design Food Delivery App		
Design Movie Ticket Booking System		
Design URL Shortener		
Design Logging Framework		
Design Rate Limiter		

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	Fresh

Revised	**Fresh**