

12 System Design questions for SDEs *and how to answer them*



Introduction

Learning System Design completely on your own can be a daunting task.

There's a seemingly never-ending list of things to learn in the path to mastery.

In this course, you'll see what are the most important parts of system design, and how to build a solid knowledge base.

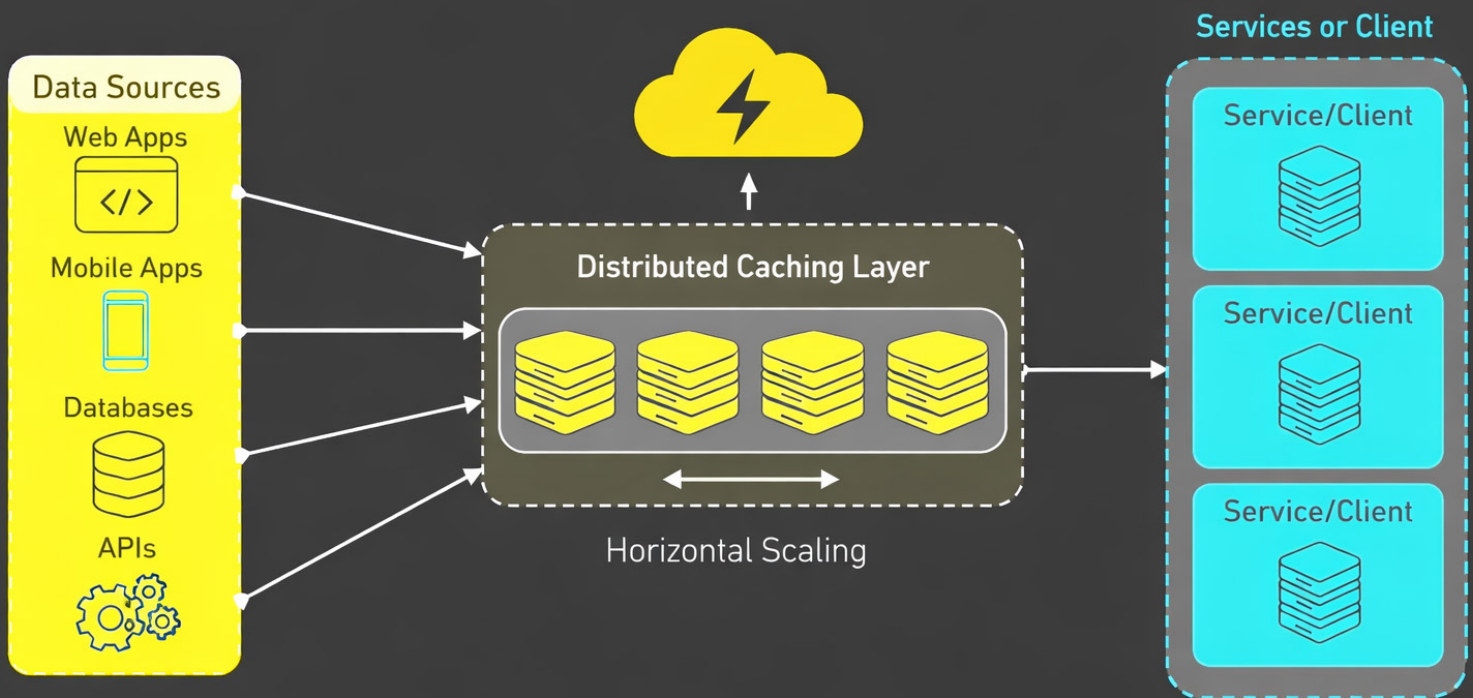
We hope this course inspires you to dive deeper into the world of distributed systems.



List of Questions

1. Distributed Caching Service
2. Distributed Rate Limiter
3. Distributed Event Bus
4. Data Processing System
5. Chat Messenger App
6. Location Based Service
7. Live Streaming Service
8. Online Gaming App
9. AuthN / AuthZ Service
10. Social Media News Feed
11. Ecommerce Platform
12. AI-Chat System





1. Design a highly-scalable Distributed Caching Service

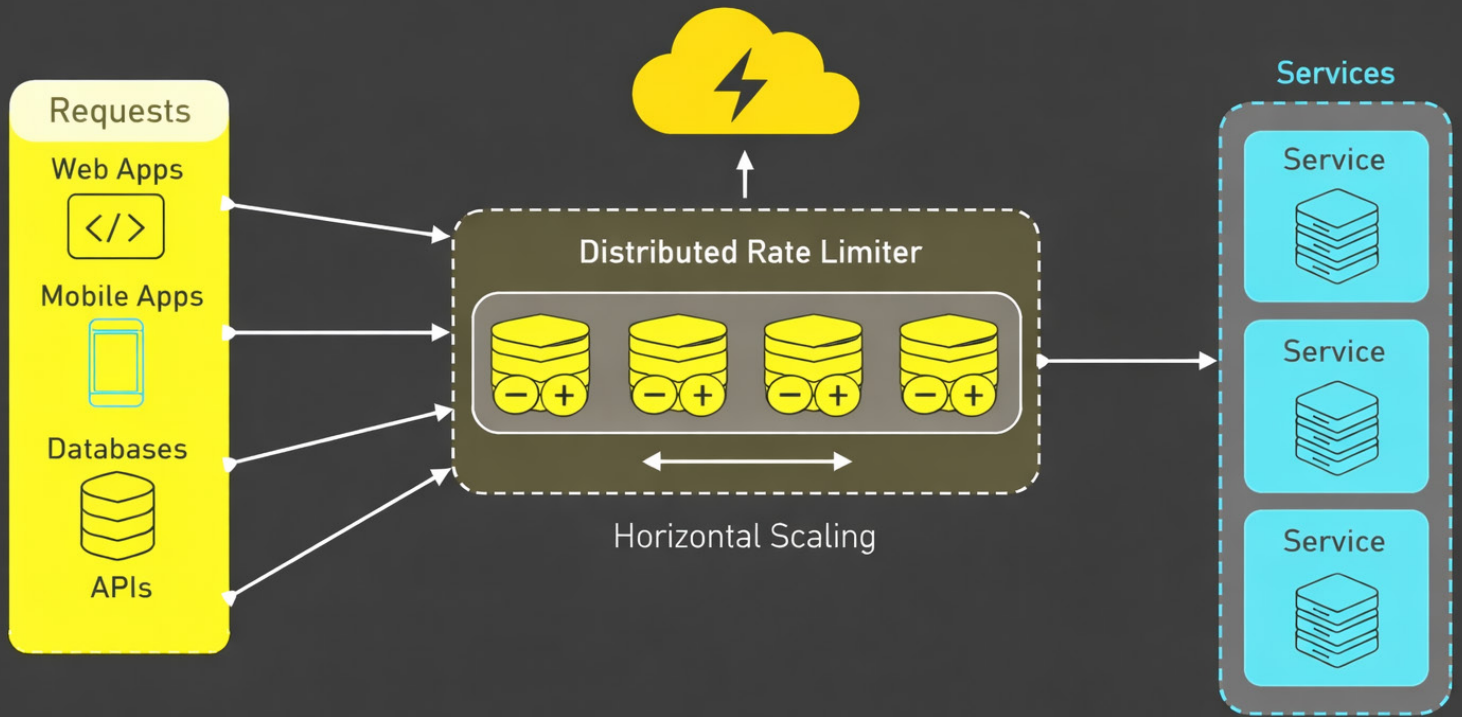
Why it matters:

- AI product
- Cost explosion risk
- Latency sensitive

What it tests:

- Request flow for AI inference
- Similarity Caching
- Vector math





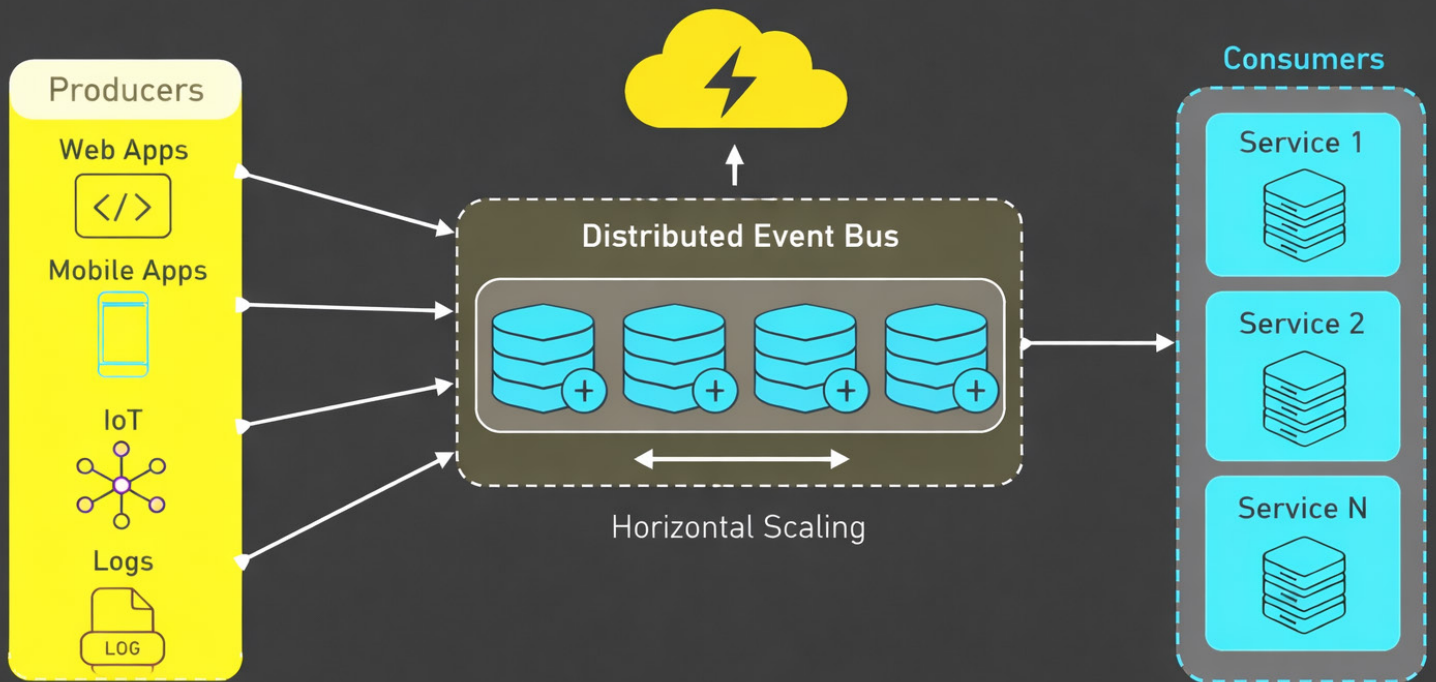
2. Design a low-latency Distributed Rate Limiter

Why it matters:

- Real production problem
- Used everywhere
- Forces tradeoffs

What it tests:

- Algorithms (token bucket, sliding window, etc.)
- Distributed consistency
- Latency vs accuracy



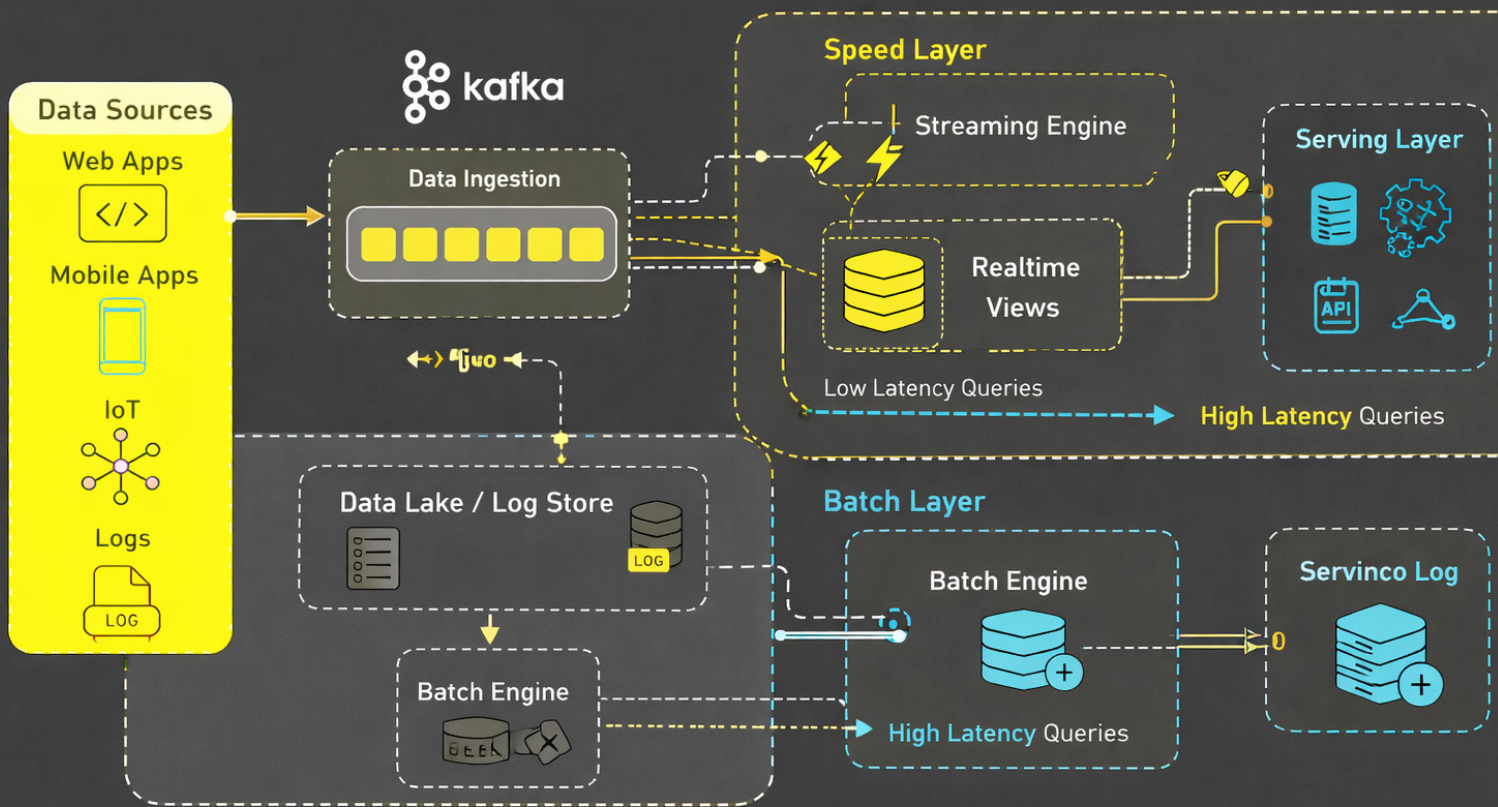
3. Design a highly-scalable Distributed Event Bus

Why it matters:

- Backbone for microservices
- Ordering guarantees
- System decoupling

What it tests:

- Partitioning Strategy
- Async Processing Management
- Distributed Failure Management



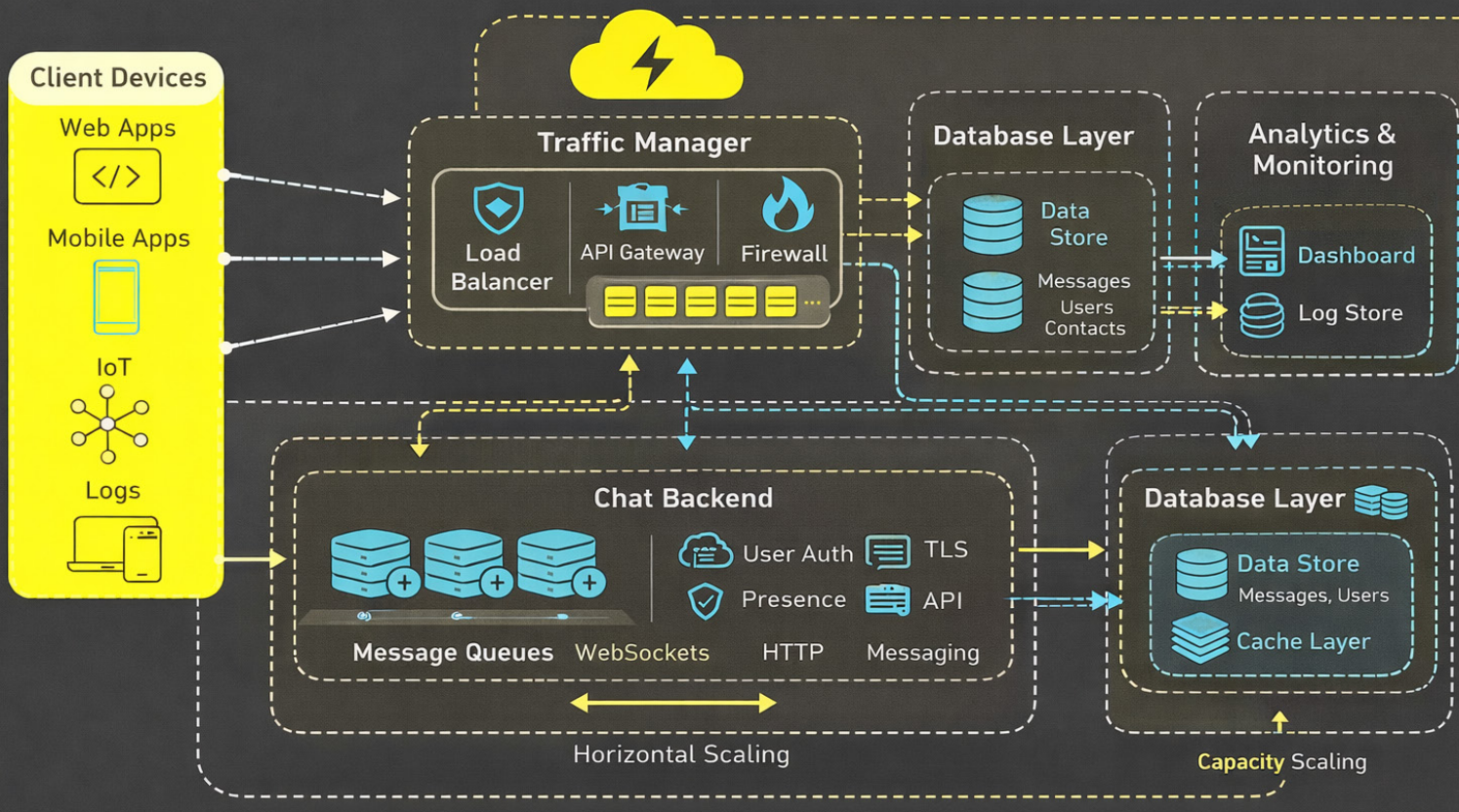
4. Design a highly-scalable Data Processing System

Why it matters:

- Real-time insights
- Cost control
- Data reliability

What it tests:

- Batch vs. stream processing
- Backpressure handling
- Data engineering fundamentals



5. Design a highly-scalable Chat Messenger App

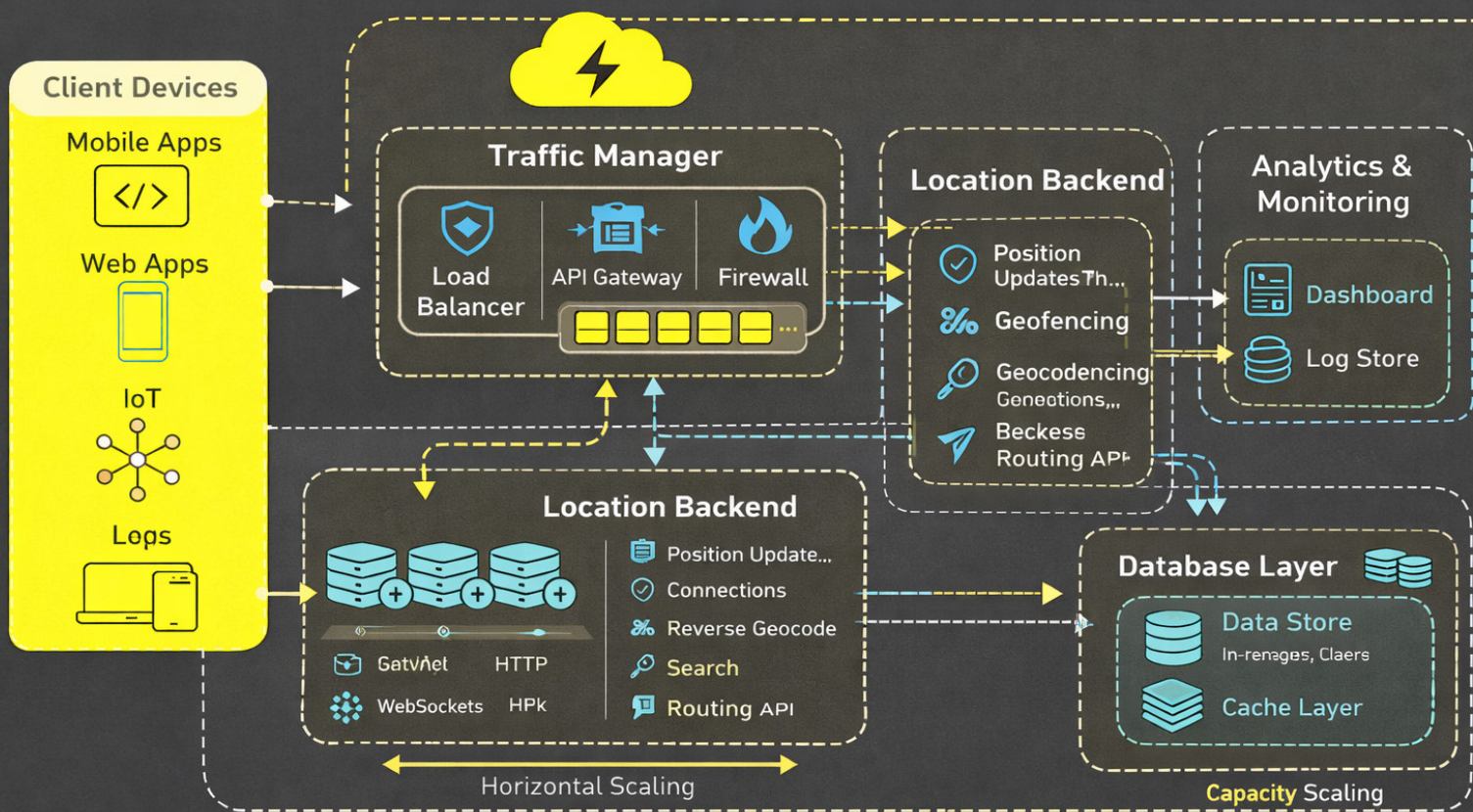
Why it matters:

- Globally distributed system
- Real-time communication
- Message Processing

What it tests:

- Message ordering
- Delivery guarantees
- Group messaging scalability





6. Design a highly-scalable Location Based Service

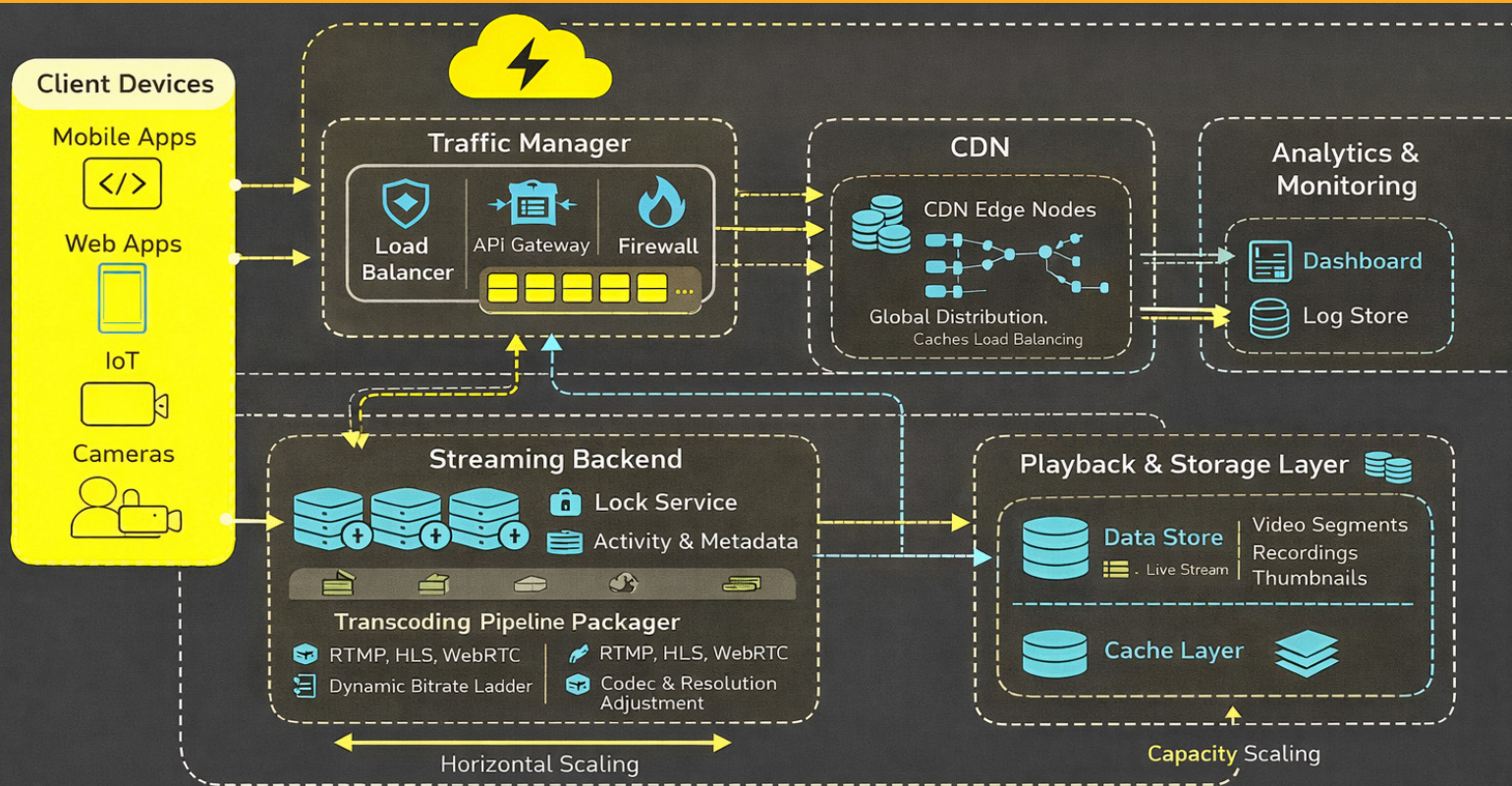
Why it matters:

- Fundamental to businesses like Maps, Delivery, etc.
- High request volume
- Latency sensitive

What it tests:

- Geo indexing
- Proximity queries
- Consistency tradeoffs





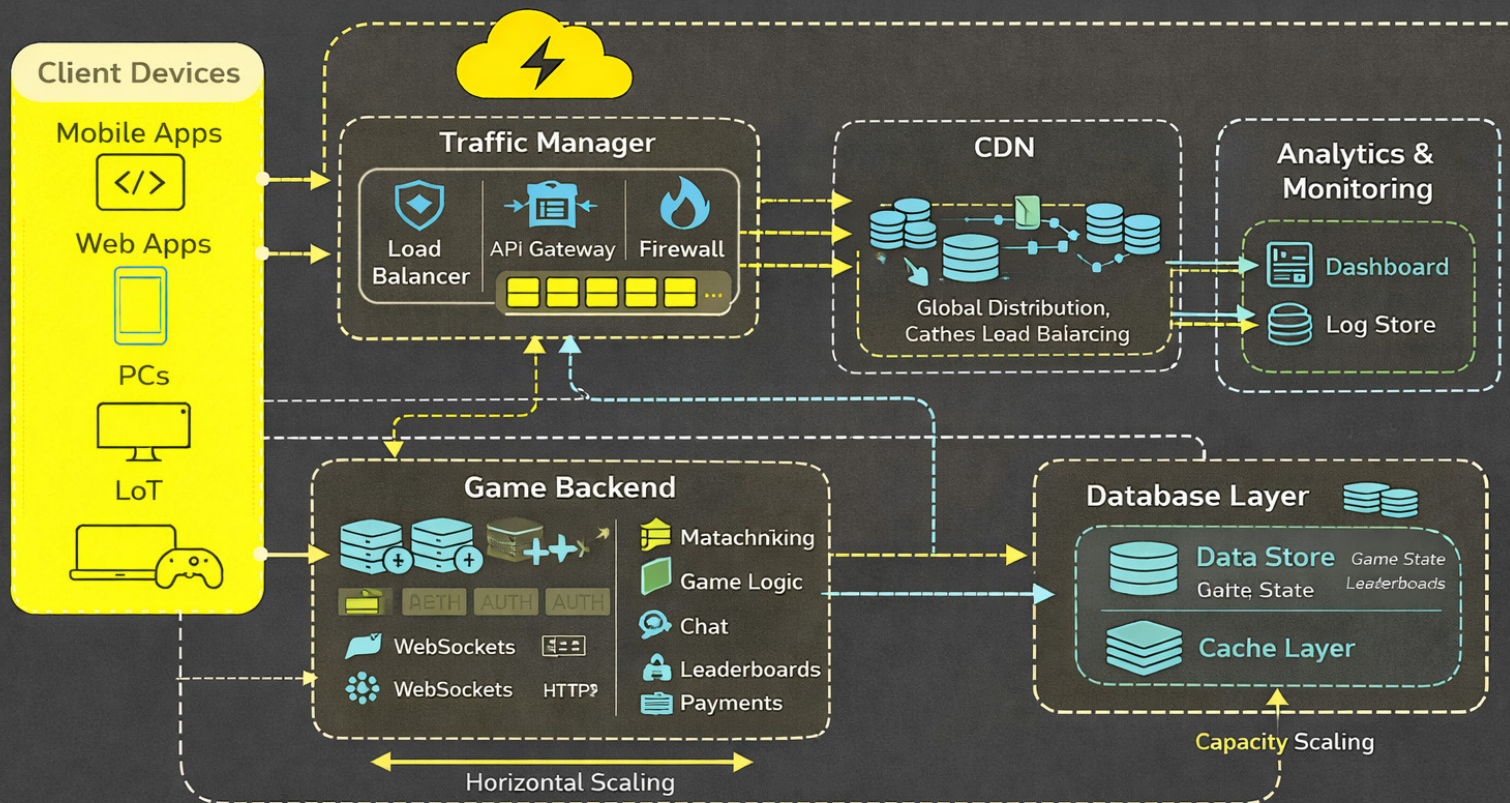
7. Design a highly-scalable Live Streaming Service

Why it matters:

- Massive bandwidth requirement
- Real-time latency
- Multiple network protocols to choose from

What it tests:

- Protocol knowledge (WebRTC, DASH, etc...)
- Fan-out delivery
- CDN fundamentals



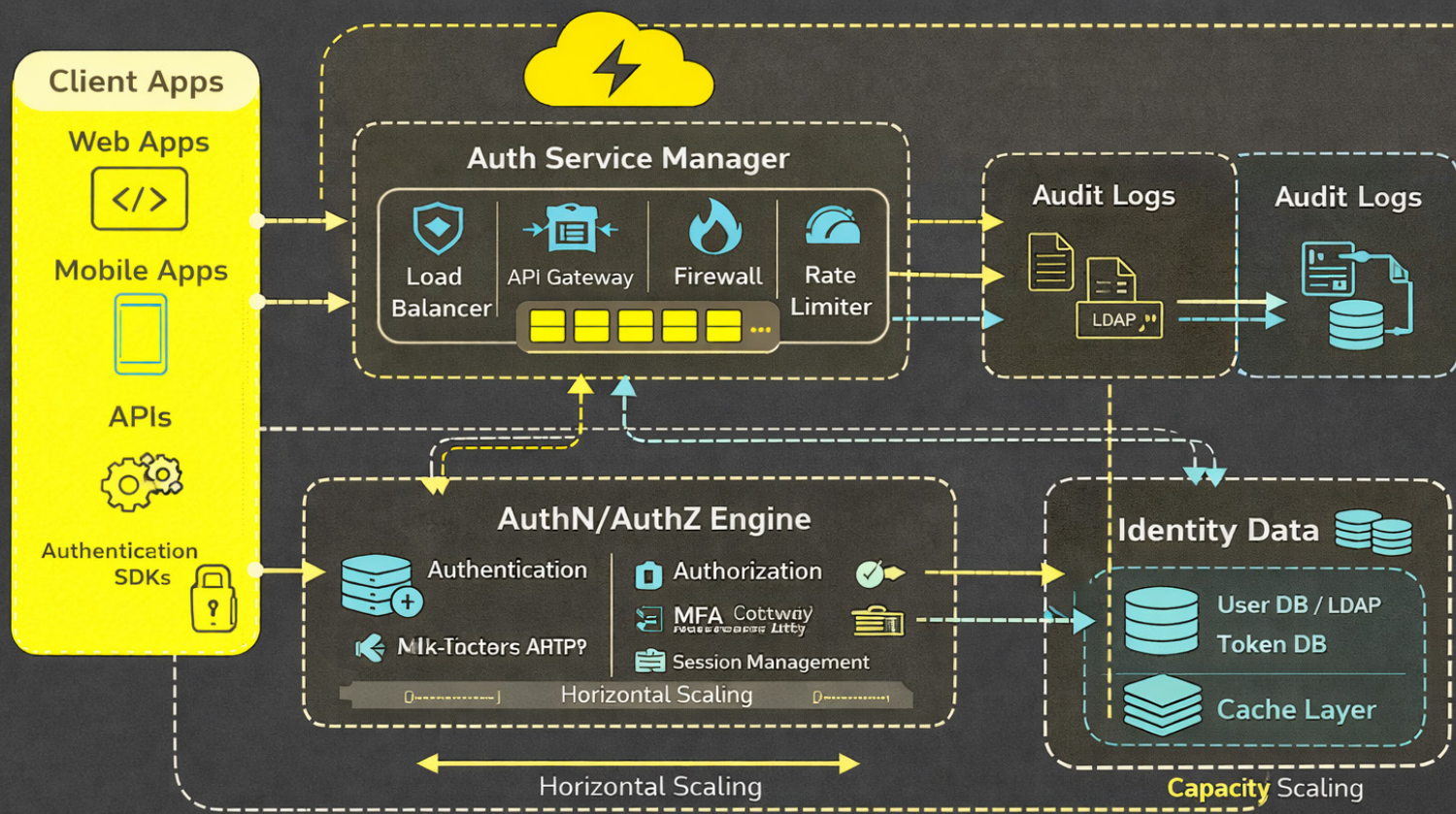
8. Design a highly-scalable Online Gaming App

Why it matters:

- Massive concurrency
- Cheating risk

What it tests:

- State synchronization
- Latency Minimization



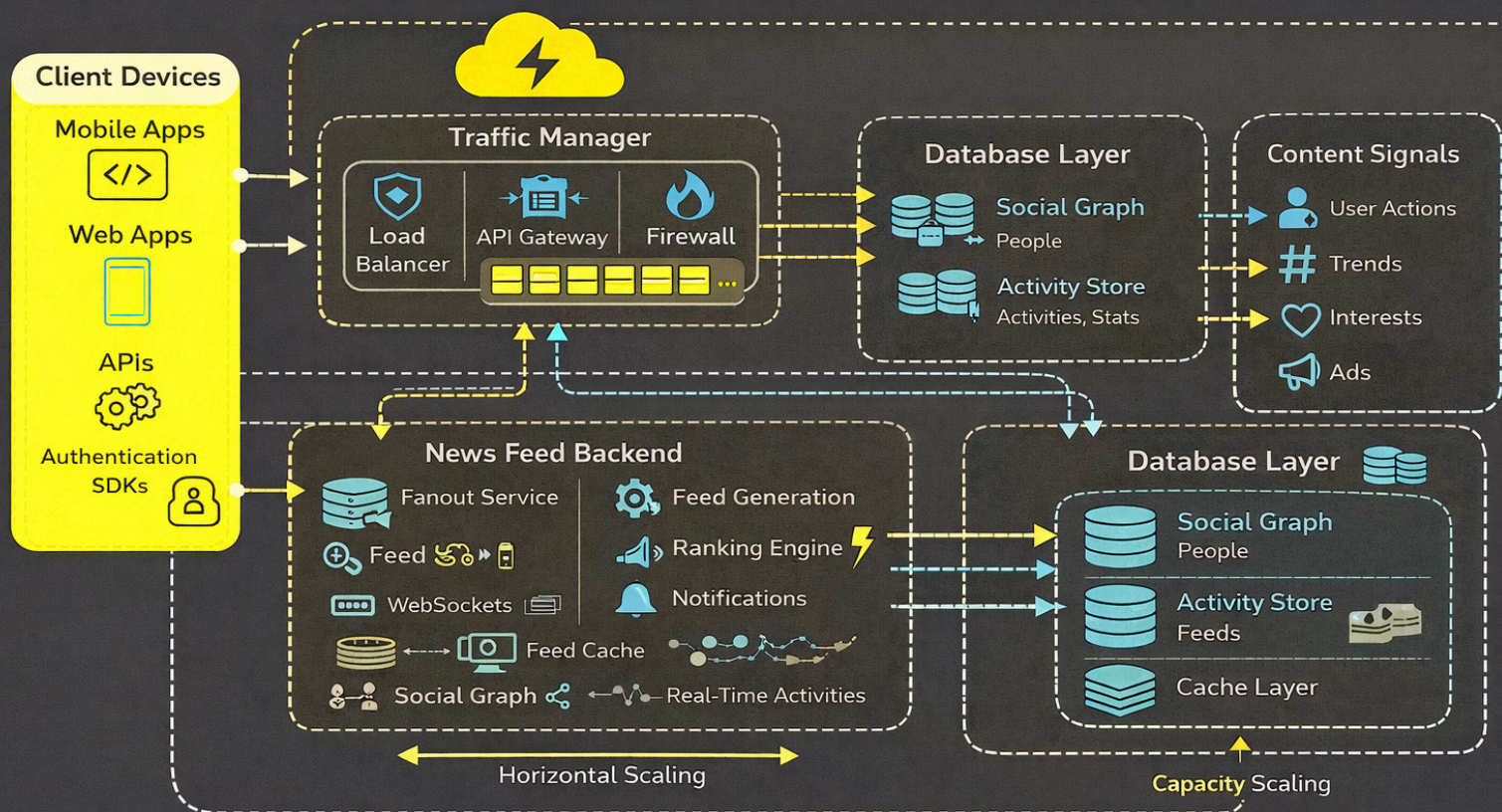
9. Design a low-latency AuthN / AuthZ Service

Why it matters:

- Security fundamentals
- Used everywhere

What it tests:

- Security fundamentals
- Token expiry
- Permission modeling



10. Design a highly-scalable Social Media News Feed

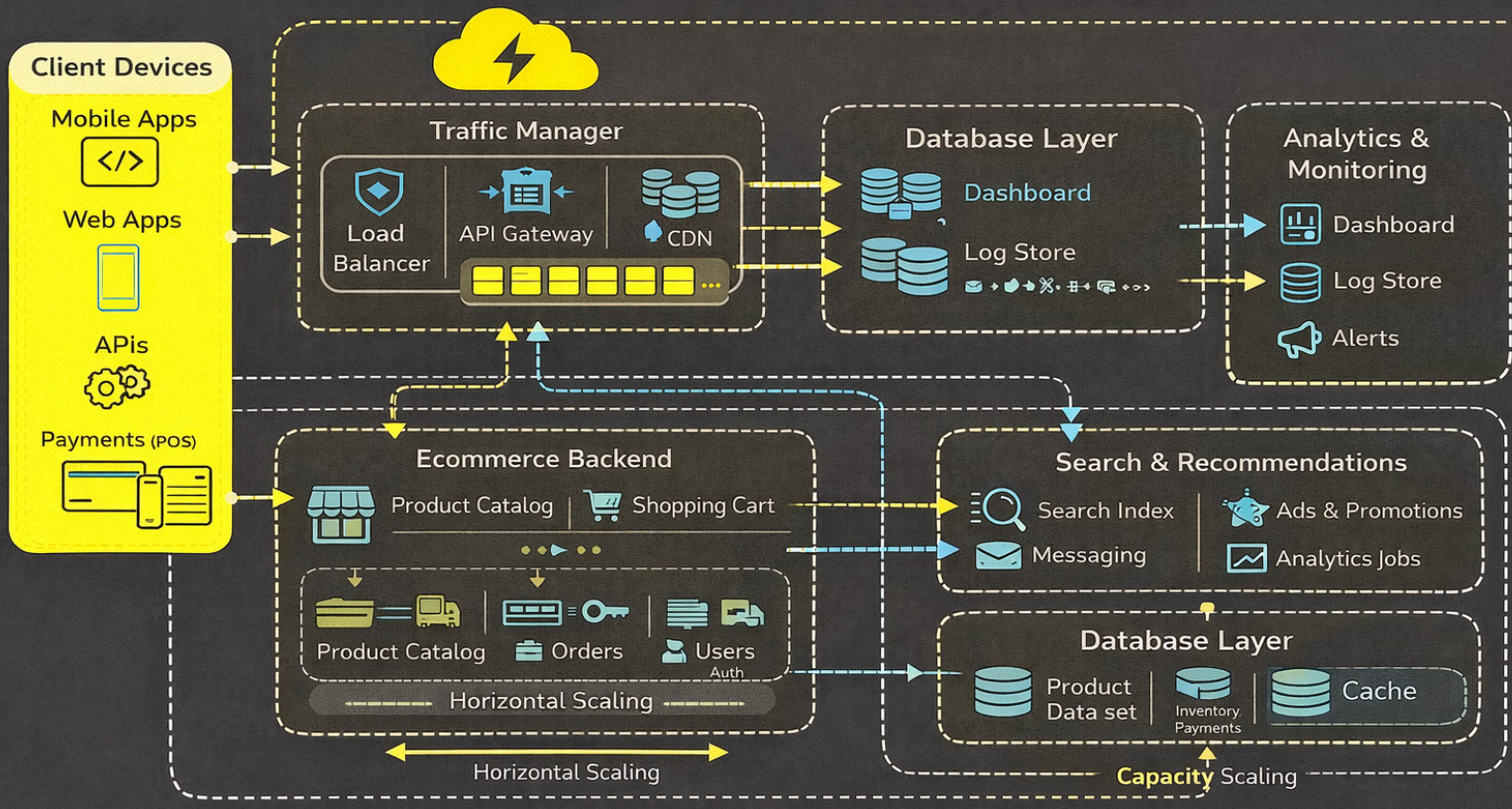
Why it matters:

- Read-heavy scale
- Content relevance
- Real-time recommendations

What it tests:

- Push vs. Pull
- Ranking strategy
- Caching strategies





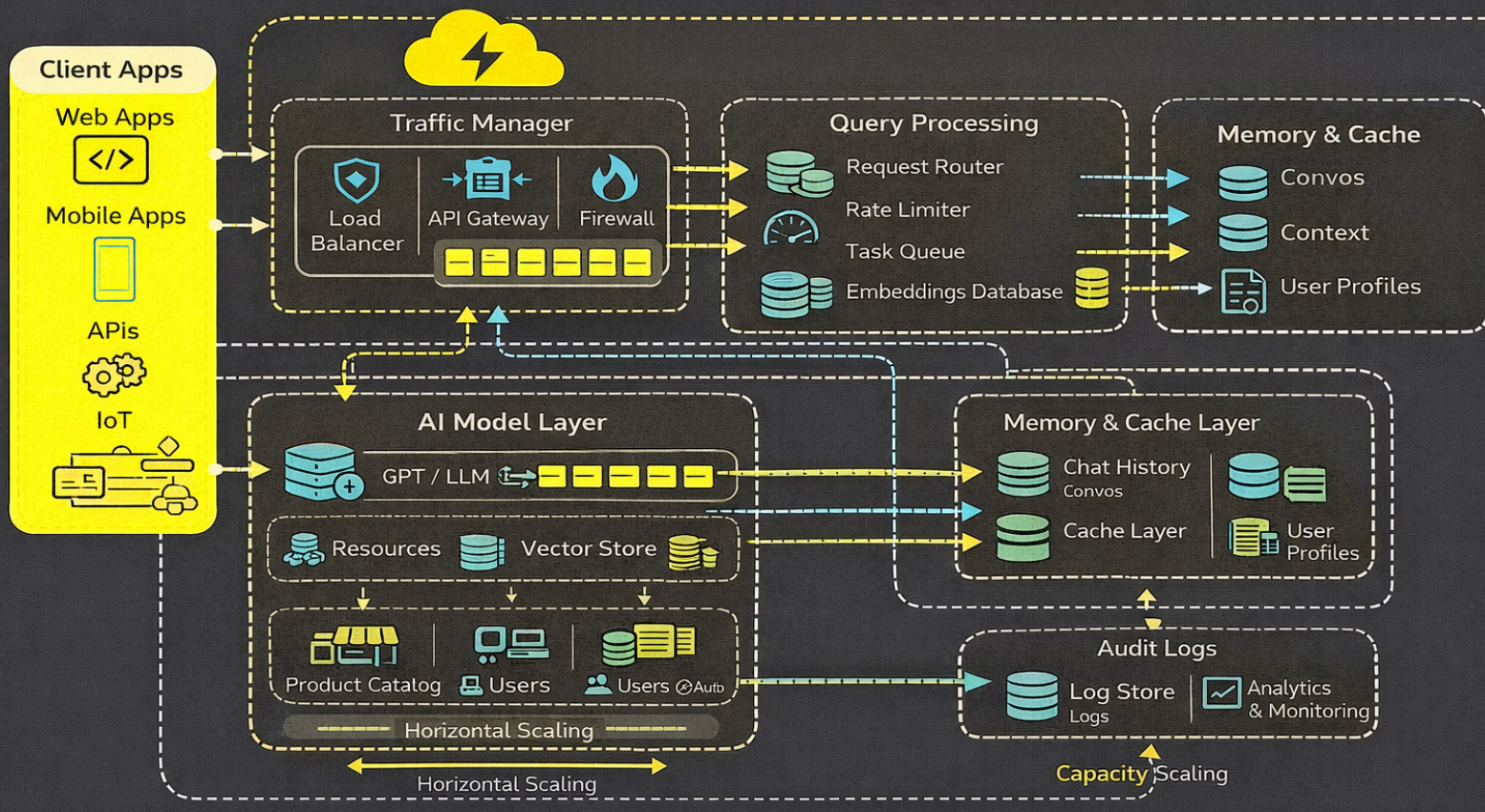
11. Design a highly-scalable Ecommerce Platform

Why it matters:

- Peak traffic events
- Inventory Management
- Order placement and flow

What it tests:

- Transaction integrity
- Consistency tradeoffs
- Order workflow design



12. Design a AI-Chat System like ChatGPT or DeepSeek

Why it matters:

- Read-heavy scale
- Content relevance
- Real-time recommendations

What it tests:

- Push vs. Pull
- Ranking strategy
- Caching strategies



What next?

This questions show you what actually matters in system design, and the thinking expected at scale.

But diagrams alone do not make you good at system design.

If you want to learn how to think like a system designer, start practicing problems the right way.

**Learn more at:
<https://interviewready.io>**