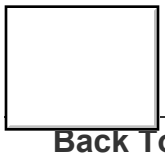


Back To Course Home

Grokking Modern System Design Interview for Engineers & Managers

0% completed

Search Course

System Design Interviews

Introduction

Abstractions

Non-functional System Characteristics

Back-of-the-envelope Calculations

Building Blocks

Domain Name System

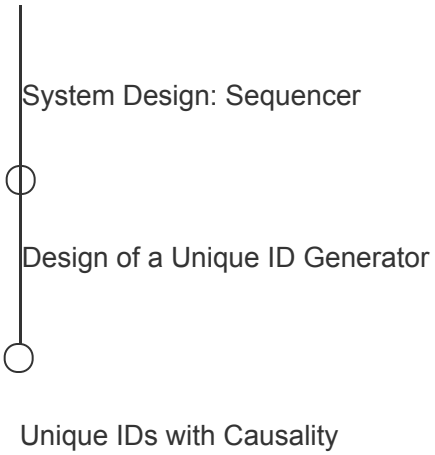
Load Balancers

Databases

Key-value Store

Content Delivery Network (CDN)

Sequencer



Distributed Monitoring

Monitor Server-side Errors

Monitor Client-side Errors

Distributed Cache

Distributed Messaging Queue

Pub-sub

Rate Limiter

- [ ]

Blob Store

- [ ]

Distributed Search

- [ ]

Distributed Logging

- [ ]

Distributed Task Scheduler

- [ ]

Sharded Counters

- [ ]

Concluding the Building Blocks Discussion

- [ ]

Design YouTube

- [ ]

Design Quora

- [ ]

Design Google Maps

- [ ]

Design a Proximity Service / Yelp

- [ ]

Design Uber

- [ ]

Design Twitter

- [ ]

Mark Course as Completed

System Design: Sequencer

Understand the basics of designing a sequencer.

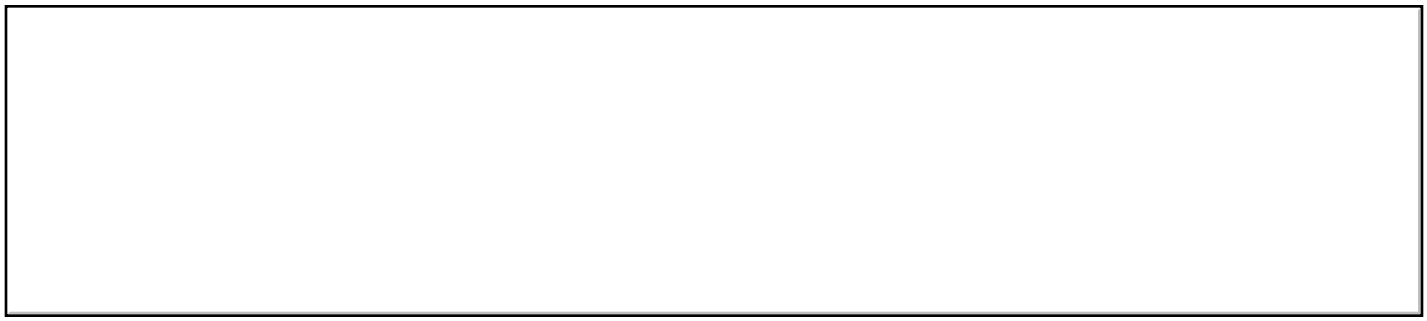
We'll cover the following

Motivation#

There can be millions of events happening per second in a large distributed system. Commenting on a post on Facebook, sharing a Tweet, and posting a picture on Instagram are just a few examples of such events. We need a mechanism to distinguish these events from each other. One such mechanism is the assignment of globally unique IDs to each of these events.

Assigning a primary key to an entry in a database is another use case of a unique ID. Usually, the auto-increment feature in databases fulfills this requirement. However, that feature won't work for a distributed database, where different nodes independently generate the identifiers. For this case, we need a unique ID generator that acts as a primary key in a distributed setting—for example, a horizontally-sharded table.

A unique ID helps us identify the flow of an event in the logs and is useful for debugging. A real-world example of unique ID usage is Facebook's end-to-end performance tracing and analysis system, Canopy. **Canopy** uses TraceID to identify an event uniquely across the execution path that may potentially perform hundreds of microservices to fulfill one user request.



Assigning a unique TraceID to each event

How do we design a sequencer?#

We've divided the sequencer's comprehensive design into the following two lessons:

1. **Design of a Unique ID Generator:** After enlisting the requirements of the design, we discuss three ways to generate unique IDs: using UUID, using a database, and using a range handler.
2. **Unique IDs with Causality:** In this lesson, we incorporate an additional factor of time in the generation of IDs and explain the process by taking causality into consideration.

Unique IDs are important for identifying events and objects within a distributed system. However, designing a unique ID generator within a distributed system is challenging. In the next lesson, let's look at the requirements for a distributed unique ID generation system.

Quiz on CDN's Design

Next

Design of a Unique ID Generator

Mark as Completed

Report an Issue