System Design and Architecture

System Design Fundamentals

- Scalabiltiy
- Availability
- Consistency - synchronization across components
- Latency
- Throughput

High-Level System Design

- Requirements - users, traffic, constraints
- Components - Core Services, Databases, APIs
- Data Storage - SQL vs NoSQL Databases
- Communication - REST, gRPC, WebSockets, Messaging Queues
- Handle Scalability & Load - Load balancing, caching, partitioning
- Security Considerations - Auth, encryption, DDoS protection

DB and Storage

- SQL (relational) vs Non-SQL (non-relational)
  - SQL: ACID compliance, Structured Queries (MySQL, PostgreSQL)
    - Atomicity, Consistency, Isolation, Durability
  - NoSQL: High scalability, flexible schema (MongoDB, Cassandra)
- Sharding: distributing data across multiple db's
- Replication: Copying data for high availbility
- Indexing: Improves query performance

Load Balancing & Caching

- Load balancers: distribute traffic accross servers (Nginx, AWS ALB)
- CDN (Content Delviery Network): Improves load times for static assets
- Caching Strategies:
  - Client-Side Caching (Browser, CDN)
  - Server-side Caching (Redis, Memached)
  - Database Caching (Query caching, materialized views)

Messages, Queues, & Event Driven Architecture

- Kafka, RabbitMQ, AWS SQS for asynchronous processing
- Pub/Sub Model. Used for decoupling microservices
  - Publishers send messages to a topic, Subscribers listen to topics and receive messages when available.

Microservices & API Design

- Monolithic vs Microservices
  - Monolithic - easier to develop, harder to scale

  - Microservices - Scalable, fault-tolerant, but complex
- API Design Principles
  - RESTful APIs: Stateless, uses HTTP methods
  - gRPC: High-performance, uses Protobufs
  - WebSockets: Real-time bidirectional communication

Security Considerations

- Authentication & Authorization: OAuth, JWT, API keys
- Data Encryption: At rest (AES) and in transit (TLS/SSL)
- DDoS Protection: Rate limiting, WAF (Web Application Firewall)
- Input Validation: Preventing SQL Injection, XSS, CSRF attacks

Scalability & High Availability

- Vertical Scaling (Scaling Up): Increasing CPU/RAM on a single server
- Horizontal Scaling (Scaling Out): Adding More Servers

Common System Design Questions

1. Design a URL Shortener
2. Design a Rate Limiter for APIs
3. Design a Distributed File Storage System
4. Design a Real-Time Messaging App
5. Design a Recommendation System

** Data Engineering Services & Technologies Cheat Sheet**

Category	Technologies/Services	What to Focus On
Data Ingestion (Batch & Streaming)	Kafka, Pub/Sub, Kinesis, Flume	How streaming data pipelines work, event-driven architecture
Data Processing (Batch)	Apache Spark, Hadoop MapReduce, Dataflow	How distributed processing works, optimizations
Data Processing (Streaming)	Kafka Streams, Apache Flink, Apache Beam	How real-time event processing works
ETL Orchestration	Apache Airflow, AWS Step Functions, Prefect	DAG scheduling, retry mechanisms, dependencies
Data Storage (Cloud & On-Prem)	Amazon S3, Google Cloud Storage, HDFS	How to store large-scale data efficiently
Data Warehousing	BigQuery, Snowflake, Amazon Redshift	Columnar storage, partitioning, indexing for fast queries
NoSQL Databases	DynamoDB, Cassandra, MongoDB, Bigtable	Trade-offs vs. SQL, use cases for unstructured data

Category	Technologies/Services	What to Focus On
SQL Databases	PostgreSQL, MySQL, SQL Server	Query optimization, indexing, partitioning
Message Queues & Pub/Sub	Kafka, RabbitMQ, AWS SNS/SQS	How pub/sub systems work, async processing
Data Modeling	Star Schema, Snowflake Schema, OLAP vs. OLTP	Best practices for analytical queries
Microservices Communication	REST APIs, gRPC, WebSockets, Message Queues	When to use each for data pipelines
Big Data File Formats	Parquet, Avro, ORC, JSON, CSV	How columnar formats improve performance
Cloud Platforms	AWS (S3, Glue, Lambda), GCP (BigQuery, Dataflow)	How these services fit into data pipelines
Monitoring & Logging	Prometheus, Grafana, CloudWatch	How to track data pipeline performance
Machine Learning in Data Pipelines	Spark ML, TensorFlow, Kubeflow	ML model deployment in data workflows (optional)