xBasic System Design Questions

1. **Question:** What is system design, and why is it important?
   - **Answer:** System design is the process of defining the architecture, components, and interactions of a system to meet specified requirements. It's important because it ensures the system is scalable, reliable, and meets user needs efficiently.
2. **Question:** Explain the concept of scalability in system design.
   - **Answer:** Scalability refers to the ability of a system to handle increased workload or growth without compromising performance or reliability. It involves designing components and architectures that can scale horizontally (adding more instances) or vertically (increasing resources).
3. **Question:** Describe the difference between SQL and NoSQL databases.
   - **Answer:** SQL databases (e.g., MySQL, PostgreSQL) are relational and use structured query language for data manipulation. NoSQL databases (e.g., MongoDB, Redis) are non-relational and offer flexibility in data modeling and horizontal scalability.

Intermediate System Design Questions

4. **Question:** How would you design a URL shortening service like Bitly?
   - **Answer:** Discuss components like URL shortening algorithm, database schema for storing mappings, scalability considerations using distributed databases or caching, and handling of analytics and metrics.
5. **Question:** What are microservices, and why would you use them in system design?
   - **Answer:** Microservices are small, independent services that communicate over well-defined APIs. They allow for modular development, scalability of individual components, and flexibility in technology choices. They're beneficial for complex systems requiring flexibility and resilience.
6. **Question:** Explain the concept of RESTful APIs and their advantages.
   - **Answer:** RESTful APIs use HTTP methods (GET, POST, PUT, DELETE) to manipulate resources. They are stateless, scalable, and promote loose coupling between clients and servers. Advantages include simplicity, interoperability, and ease of caching.

Advanced System Design Questions

7. **Question:** Design a fault-tolerant system for a real-time chat application.
   - **Answer:** Discuss components like message brokers (e.g., Kafka), distributed databases for message storage, redundant servers for high availability, and strategies for handling network partitions and node failures.
8. **Question:** Compare and contrast the benefits of using an event-driven architecture versus a queue-based architecture.
   - **Answer:** Discuss communication patterns, scalability, and use cases for each architecture. Highlight scenarios where real-time updates (event-driven) versus sequential processing (queue-based) are beneficial.
9. **Question:** How would you design a recommendation system for an e-commerce platform like Amazon?
   - **Answer:** Outline components like user profiling, item catalog, recommendation algorithms (collaborative filtering, content-based filtering), real-time updates, and scalability considerations for handling large datasets and concurrent users.