Java DSAlgo TI2 script

> *This script is created to conduct the second round of Technical Interviews for candidates marked by Java DSAlgo SkillSpecs, using the DSAlgo TI2 template*

Candidate's requirements and preparedness

Here you can find instructions for our candidates.

It contains information about preparing the candidate for the interview, the knowledge requirements, and how to prepare the workplace.

This information is valuable for both sides, for the interviewer as well as for the candidate, because the interviewer can check what to expect from the candidate.

Interviewer's scripts

Introduction:

1. **Welcome and Introduction:**
    - Greet the candidate warmly.
    - Introduce yourself and mention your role in the company.
2. **Overview of the Interview:**
    - Briefly explain the interview structure (e.g., technical questions, behavioral questions, problem-solving exercises).

Review Candidate's Background (5-10 minutes):

- Ask the candidate to introduce themselves. (Pay attention to structure and self-presentation)
- Explore the candidate's technical skills and experiences.
- Ask about specific projects and their role in them. (Pay attention to "drivers" and ownership skills)

Technical Questions:

1. **Ask about topics from the DSAlgo TI2 template, focusing on the following topics**

    Distributed systems / Microservices

    - Event-driven
    - Circuit breaker
    - Saga
    - CQRS

    Data storage / Data model design

    - Indexing, partitioning, CAP(PACELC), normalization vs denormalization
    - NoSQL: types and use cases

    API Design

    - What makes a good API design, and why is it important?
    - How can RESTful principles be applied to design scalable and maintainable APIs?

    Monitoring and Logging

    - Why is monitoring crucial in system design, and what tools and techniques are available for system monitoring?
    - How can logging be effectively implemented to facilitate debugging and system analysis?

    Continuous Integration/Continuous Deployment (CI/CD)

    - How does CI/CD contribute to effective system design and development workflows?
    - What are the key principles of CI/CD, and how can they be implemented?

    Deployment Strategies

    - Explore different deployment strategies, such as blue-green deployment, canary release, and rolling deployment.

- How do these strategies contribute to system availability and maintainability?

Spring Framework / Spring Boot

- Design patterns implemented in Spring (Proxy, Factory, Strategy, Chain of responsibility)
- Spring Boot autoconfiguration and starters

Optional: Cloud (Amazon Web Services)

- SQS, SNS
- DynamoDB, Aurora
- Lambdas
- System design and analysis

Additional topics to discuss:

- **Microservices vs. Monolith:**
  - Compare and contrast microservices architecture with monolithic architecture.
  - When is it appropriate to choose a microservices architecture, and what are the benefits and challenges associated with it?
- **Databases and Data Storage:**
  - How do you choose the right database for a given application? Consider relational databases, NoSQL databases, and in-memory databases.
  - Discuss strategies for data modeling, normalization, and denormalization.
- **Scalability and Performance:**
  - What are the key considerations for designing a scalable system?
  - How can systems be optimized for performance, and what role do caching and load balancing play in scalability?
- **Security in System Design:**
  - What are the fundamental principles of designing a secure system?
  - How can you protect against common security threats, such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF)?
- **Fault Tolerance and Reliability:**
  - How can a system be designed to be fault-tolerant and reliable?
  - Discuss strategies for handling failures, implementing redundancy, and ensuring system availability.
- **Communication Protocols:**
  - Explore different communication protocols used in system design, such as HTTP/HTTPS, WebSocket, and message queues.
  - How do these protocols contribute to the efficiency and responsiveness of a system?
- **Concurrency and Parallelism:**
  - Discuss the challenges and solutions related to handling concurrency in a distributed system.
  - How can parallelism be leveraged for performance improvements?
- **Caching Strategies:**
  - What role does caching play in system design, and what are common caching strategies?
  - How can caching be used to improve response times and reduce the load on databases?
- **Cost Optimization:**
  - How can system design impact the cost of infrastructure and operations?
  - Discuss strategies for optimizing costs, especially in cloud-based architectures.
- **Elasticity and Auto-scaling:**
  - What is elasticity in the context of system design, and how does auto-scaling work?
  - How can systems dynamically adapt to varying workloads to ensure optimal resource utilization?

2. **System Design Exercise 1 (30 minutes):**
   - Introduce the Problem statement to the candidate providing minimum information.
   - Allow the candidate to clarify the details, ask additional questions, work through the problem, and explain their thought process.
   - Add some complexity to the problem statement during the conversation

Sample 1: URL Shortening Service

Problem: Design a URL shortening service like Bitly.

Solution:

- Use a relational database to store mappings between short and long URLs.
- Generate a short URL using a unique identifier or a hashing algorithm.
- Implement a caching mechanism for frequently accessed URLs.
- Ensure scalability and availability by distributing the service across multiple servers.

Sample 2: Social Media Feed System

Problem: Design a system to generate and display a user's social media feed.

Solution:

- Use a microservices architecture with services for user management, post creation, and feed generation.
- Store user posts in a scalable and distributed database.
- Implement a feed generation algorithm based on relevance, recency, and user interactions.
- Use a caching layer to optimize feed retrieval.

Sample 3: Online Marketplace

Problem: Design an online marketplace where users can buy and sell products.

Solution:

- Use a microservices architecture with services for user management, product listing, order processing, and payment.
- Implement a search and recommendation engine for product discovery.
- Ensure secure and reliable payment processing using a payment gateway.
- Use a scalable database to store product information and user transactions.

These sample problems and solutions cover a range of difficulty levels and demonstrate both algorithmic problem-solving skills and system design considerations. Keep in mind that real-world scenarios may involve more complexity and trade-offs.

Candidate Questions:

1. **Invite Candidate Questions (5 minutes):**
    - Allow the candidate to ask questions about the company, team, or role.
    - Provide information about the company culture and expectations.

Closing:

1. **Next Steps:**
    - Explain the next steps in the hiring process.
    - Clarify the timeline for feedback and potential follow-up interviews.
2. **Thank You:**
    - Express gratitude for the candidate's time and participation.
    - Provide contact information for any follow-up questions.

Please **watch the video** to learn about DSAlgo TI specifics: