**2-Day Advanced Preparation Plan for R1 Interview (1st Feb 2025)**

**Role Requirement: 3-6 Years Experience**

**Mandatory Skills: Java, Spring Boot, Microservices, Hibernate, Basic SQL**

**Good to Have: Angular / Any UI Framework**

**📅 Day 1: Thursday, 30th Jan 2025**

**Morning (8:00 AM - 12:00 PM) - Advanced Java Concepts**

**🔹 Topics to Cover:**

* **Concurrency & Multithreading:** ThreadLocal, ForkJoinPool, CompletableFuture
* **JVM Performance Tuning:** Heap & Stack, GC tuning, Profiling tools
* **Design Patterns:** Factory, Strategy, Observer, Circuit Breaker in Microservices
* **Java 8-17 Enhancements:** Sealed classes, Records, Virtual Threads

**🔹 Interview Questions:**

1. How does ForkJoinPool work in Java?
2. What is ThreadLocal and when should you use it?
3. How do you optimize Java applications for high performance?
4. Explain the difference between WeakReference, SoftReference, and PhantomReference.
5. What are the benefits of using Records in Java 14+?

**🔹 Problems to Solve:**

1. Implement a thread-safe singleton pattern.
2. Write a Java program to execute multiple async tasks using CompletableFuture.
3. Implement a real-time producer-consumer problem using BlockingQueue.

**Afternoon (1:00 PM - 4:00 PM) - Spring Boot & Microservices**

**🔹 Topics to Cover:**

* **Spring Boot Deep Dive:** Lazy vs Eager initialization, Actuator, Metrics
* **Spring Security:** OAuth2, JWT, CSRF, RBAC
* **Microservices Communication:** Synchronous (REST, Feign) vs Asynchronous (Kafka, RabbitMQ)
* **Resilience Patterns:** Circuit Breaker (Resilience4j), Retry Mechanisms
* **Dockerizing Spring Boot Apps & Deployment**

**🔹 Interview Questions:**

1. How do you secure a REST API using JWT in Spring Boot?
2. What is the difference between Feign Client and RestTemplate?
3. Explain the Circuit Breaker Pattern and how to implement it using Resilience4j.
4. How do you manage transactions across microservices?
5. What are the benefits of using Kafka in Microservices?

**🔹 Problems to Solve:**

1. Create a secured Spring Boot REST API using JWT authentication.
2. Implement a Feign Client to call another microservice.
3. Configure a Spring Boot application with Resilience4j for Circuit Breaking.

**Evening (6:00 PM - 9:00 PM) - Hibernate & SQL Optimization**

**🔹 Topics to Cover:**

* **Hibernate Performance Tuning:** Caching (First-Level, Second-Level, Query Cache)
* **Transaction Management:** Isolation Levels, Propagation, ACID Compliance
* **SQL Optimization:** Indexing, Query Execution Plan, Partitioning
* **Database Scaling:** Sharding, Read-Replicas, Connection Pooling

**🔹 Interview Questions:**

1. How does Hibernate’s second-level cache work?
2. What is the difference between optimistic and pessimistic locking?
3. How do you handle lazy loading exceptions in Hibernate?
4. What are the different types of indexes in SQL?
5. How would you optimize a slow-running SQL query?

**🔹 Problems to Solve:**

1. Write an HQL query to fetch employees with the second-highest salary.
2. Implement a Many-to-Many mapping in Hibernate.
3. Optimize a SQL query using indexing and analyze the execution plan.

**📅 Day 2: Friday, 31st Jan 2025**

**Morning (8:00 AM - 12:00 PM) - System Design & Architecture**

**🔹 Topics to Cover:**

* **Scalability Patterns:** Load Balancing, API Gateway, Rate Limiting
* **Database Selection:** SQL vs NoSQL (MongoDB, Cassandra)
* **Caching Strategies:** Redis, Memcached, EhCache
* **Event-Driven Architecture:** Kafka, RabbitMQ, SQS

**🔹 Interview Questions:**

1. How would you design a highly scalable e-commerce system?
2. What is the CAP theorem, and how does it impact database design?
3. How do you ensure consistency in a distributed microservices architecture?
4. What are different caching strategies, and when to use them?
5. How does Kafka handle message retention and partitioning?

**🔹 Hands-on Task:**

1. Design a microservices-based ride-sharing application (API structure, DB design).
2. Implement Redis caching for a Spring Boot application.
3. Write an API Gateway with rate-limiting functionality.

**Afternoon (1:00 PM - 4:00 PM) - Angular & UI (Optional but Good to Know)**

**🔹 Topics to Cover:**

* **Component Lifecycle & State Management**
* **Angular Services & Dependency Injection**
* **Reactive Forms & API Integration (HttpClient)**
* **Routing & Guards for Security**

**🔹 Interview Questions:**

1. What is the difference between template-driven and reactive forms in Angular?
2. How does Angular handle dependency injection?
3. What are Observables, and how are they different from Promises?
4. How do you optimize an Angular application for performance?
5. What is lazy loading in Angular, and how does it improve performance?

**🔹 Hands-on Task:**

1. Create a simple Angular app that fetches and displays user data from a REST API.
2. Implement route guards to restrict access to specific components.
3. Implement state management using RxJS and BehaviorSubject.

**Evening (6:00 PM - 9:00 PM) - Mock Interview & Final Review**

**🔹 Tasks:**

* **Solve at least 3 LeetCode/HackerRank problems (medium-hard level)**
* **Revisit tricky interview questions from the past 2 days**
* **Conduct a mock interview session with a friend or using an AI platform**
* **Prepare structured answers using STAR (Situation, Task, Action, Result) method for behavioral questions**

**Final Tips for R1 Interview Day (Saturday, 1st Feb 2025)**

✅ **Prioritize problem-solving over theoretical explanations**  
✅ **Think aloud while coding during the interview**  
✅ **Use proper design principles in system design questions**  
✅ **Be prepared to write optimized SQL queries on the spot**  
✅ **Keep responses structured and precise**