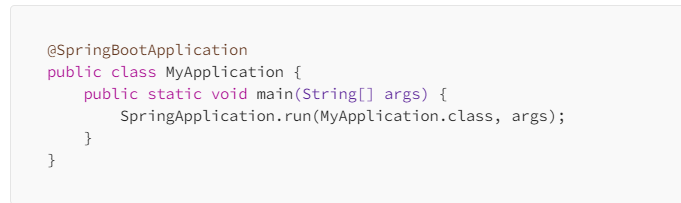
**Spring Boot and Microservice Interview Questions**

1. **What is the main class in a Spring Boot application?**

* The main class in a Spring Boot application is the entry point and is annotated with @SpringBootApplication



**2) What does the @SpringBootApplication annotation do?**

🡪 **@SpringBootApplication** is a convenience annotation that combines three annotations

**@Configuration**  : (marks the class as a source of bean definitions)

**@EnableAutoConfiguration**  : (enables Spring Boot’s auto-configuration mechanism)

**@ComponentScan**  : (scans the package of the annotated class for Spring components)

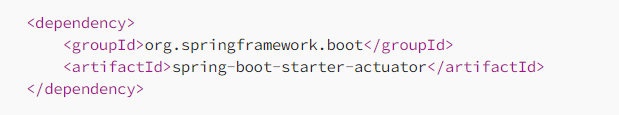
1. **How do you handle exceptions in Spring Boot?**

* You can handle exceptions in Spring Boot using @ControllerAdvice and @ExceptionHandler annotations to create a global exception handler.



1. **What is Spring Boot Actuator and what are its benefits?**

* [Spring Boot Actuator](https://javadzone.com/spring-boot-actuator-5-ways-to-boost-your-applications-performance/)provides production-ready features such as **health checks**, **metrics**, and **monitoring** for Spring Boot application.





1. **What are Spring Profiles and how do you use them?**

* [Spring Profiles](https://javadzone.com/spring-boot-profiles-mastery-5-proven-tips/) allow you to segregate parts of your application configuration and make it only available in certain environments.
* You can activate profiles using the **spring.profiles.active**property.



1. **How do you optimize the startup time of a Spring Boot application in a production environment?**

**Spring Boot provides various options for optimizing startup time. Key strategies include:**

1. **Lazy Initialization :** Use **spring.main.lazy-initialization=true** to delay bean initialization until needed.
2. **Profile-Specific Configuration** **:** Separate configurations per environment to avoid unnecessary loading.
3. **Component Scanning :** Restrict the scope of component scanning using @ComponentScan to only include essential

packages.

1. **Reduce Bean Creation :** Avoid creating unnecessary beans during startup, especially for time-intensive services.

**By reducing the number of beans initialized upfront, you can significantly speed up application startup.**

1. **Explain the concept of Spring Boot’s @ConfigurationProperties with complex objects. How would you handle nested configurations?**

* The **@ConfigurationProperties** annotation is a powerful way to map external configuration properties into Java objects
* For complex, nested configurations, Spring Boot can handle hierarchical properties through nested classes.

A screenshot of a computer program

AI-generated content may be incorrect.

1. **What are the main challenges with distributed tracing in Spring Boot microservices, and how do you implement it?**

* Distributed tracing allows tracking requests across multiple microservices.
* The challenges include latency, proper correlation of requests, and aggregating trace data across services.

**Solution**:

* **Spring Cloud Sleuth**: Automatically instruments Spring Boot applications for distributed tracing.
* **Integration with Zipkin or Jaeger**: Use Sleuth with tools like Zipkin for trace visualization and monitoring.
* **Correlation**: Propagate TraceId and SpanId headers for cross-service correlation, ensuring traceability.

1. **What is Spring Boot’s @Retryable annotation, and how do you fine-tune it for microservices reliability?**

* The @Retryable annotation in Spring Boot allows retrying a method call in case of failure.
* This is essential for improving the reliability of services that might experience transient failures (e.g., network timeouts or database issues).



1. **How do you handle versioning in Spring Boot APIs?**

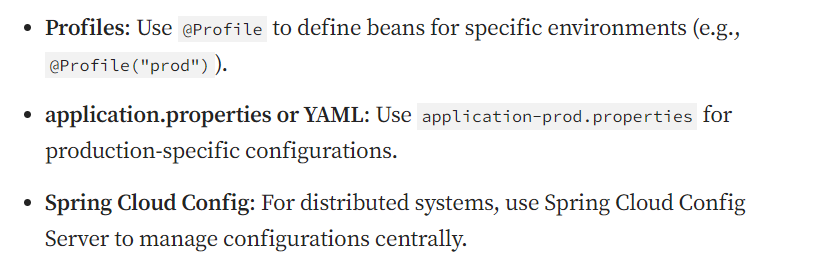
**API versioning is crucial for maintaining backward compatibility as your services evolve. Common strategies include:**

**URI Versioning** : /api/v1/resource

**Parameter Versioning** : /api/resource?version=1

**Header-based Versioning** : Through custom headers like API-Version.

1. **How do you manage external configurations in a Spring Boot application across multiple environments?**



1. **What are some common performance bottlenecks in Spring Boot applications and how do you resolve them?**
2. **Database Access**

* ✅ **Optimize Queries:** Ensure SQL queries are efficient and use proper indexes.
* 📄 **Pagination:** Fetch data in chunks using pagination to avoid memory overload.
* 🔗 **Connection Pooling:** Leverage tools like **HikariCP** for efficient database connection management.

1. **Memory Management**

* 🧠 **Avoid Memory Leaks:** Regularly review your code for unclosed resources or lingering references.
* 🔍 **Monitoring:** Use tools like **VisualVM** or **JVisualVM** to profile memory usage and identify leaks.

1. **Thread Pooling**

* ⚙️ **Thread Pool Sizing:** Tune thread pools based on system resources and expected load.
* 🧵 **Executor Services:** Use Java’s ExecutorService or Spring’s @Async with properly configured task executors for managing background tasks and HTTP request handling.

1. **Caching**

* 🚀 **Spring Caching Abstraction:** Use annotations like @Cacheable, @CachePut, and @CacheEvict to reduce database hits.
* 🗃️ **Cache Providers:** Integrate with **Ehcache**, **Caffeine**, or **Redis** for high-performance caching solutions.

1. **How do you configure and manage Spring Boot logging in production?**
2. **🔊 Logback (Default Logging Framework)**

* ✅ **Out-of-the-Box Support:** Spring Boot uses **Logback** as the default logging implementation.
* ⚙️ **Custom Configuration:** Configure advanced logging using logback-spring.xml for environment-specific setups.

1. **🌐 External Logging Integration**

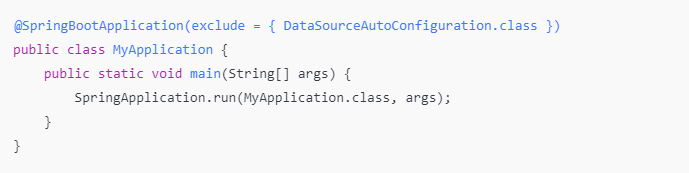
* **📦 Centralized Logging:** Integrate with logging stacks like ELK (Elasticsearch + Logstash + Kibana) or EFK (Fluentd instead of Logstash).
* **📡 Benefits:** Enables real-time log aggregation, analysis, and visualization across distributed services.

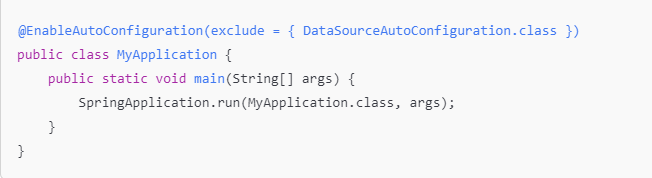
1. **🎚️ Log Levels per Environment**

* 🔧 **Environment-Specific Tuning:**
  + DEBUG – for development to get detailed application flow.
  + INFO – for production to reduce noise but keep essential logs.
* 📝 **Configuration:** Set log levels in application.yml or application.properties:

1. **How can you disable a specific auto-configuration class in Spring Boot ?**

* Disable an auto-configuration class by adding the exclude attribute:





1. **How to configure multiple datasource in spring boot application ?**

* Define Properties in application.yml or application.properties .



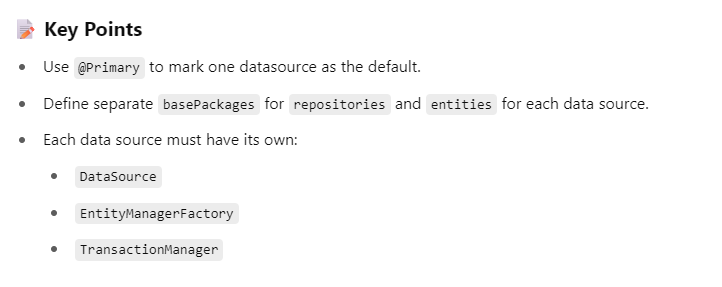
* Configure primary datasource

A screenshot of a computer program

AI-generated content may be incorrect.

Configure secondary datasource





1. **Can you explain the purpose of Stereotype annotations in the Spring Framework ?**

* Stereotype annotations in Spring are specialized annotations used to declare Spring-managed components.
* They help Spring automatically detect and register beans during component scanning.

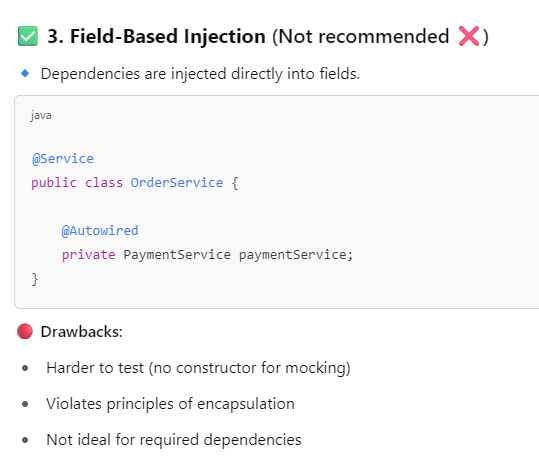
A screenshot of a chat

AI-generated content may be incorrect.

1. **How many ways we can perform dependency injection in spring or spring boot ?**

A black text on a white background

AI-generated content may be incorrect.**A white background with black text

AI-generated content may be incorrect.**A screenshot of a computer program

AI-generated content may be incorrect.****

1. **Can you provide an example of a real-world use case where @PostConstruct is particularly useful?**

* Preloading Data on Application Startup

A screenshot of a computer

AI-generated content may be incorrect.

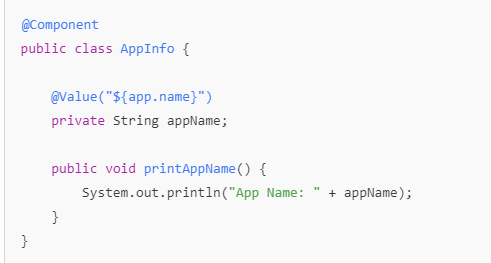


A screenshot of a computer

AI-generated content may be incorrect.

1. **How can we dynamically load values in a Spring Boot application?**

✅ **1. Using @Value Annotation (for simple values)**

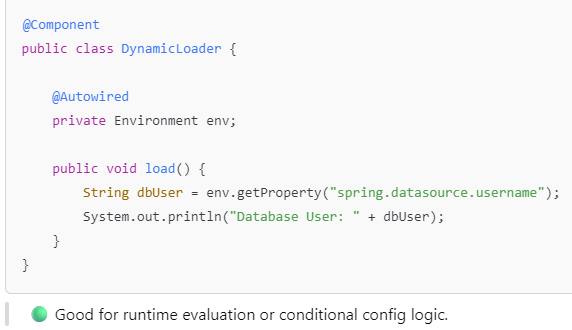


✅ **2. Using @ConfigurationProperties (for grouped values)**

**A screenshot of a computer program

AI-generated content may be incorrect.**

✅ **3. Environment Object (for runtime value resolution)**



✅ **4. External Configuration (via Environment Variables or CLI)**

A black text with numbers

AI-generated content may be incorrect.

✅ **5. Dynamic Loading from a Database / Remote Source**

* You can fetch values at runtime (e.g., for feature flags or settings) using @PostConstruct, a scheduled task, or an external config service.

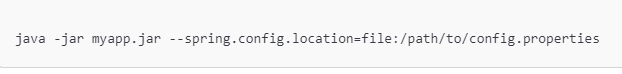
**A screen shot of a computer code

AI-generated content may be incorrect.**

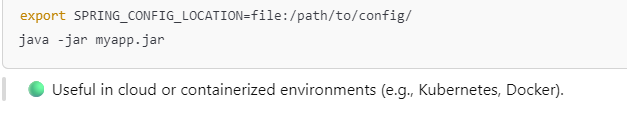
1. **How to load External Properties in Spring Boot ?**

* **Loading external properties in a Spring Boot application is a common way to separate configuration from code — especially useful for different environments (dev, test, prod).**

**✅ 1. Use --spring.config.location Command-Line Argument**

****

**✅ 2. Use SPRING\_CONFIG\_LOCATION Environment Variable**



✅ **3. Use @PropertySource in Java Config (for custom file names)**

A screenshot of a computer

AI-generated content may be incorrect.

**✅ 4. Use External Config in Spring Boot Profiles**

****

1. **Can we avoid this dependency ambiguity without using @Qualifier ?**

🔹 1. **Use @Primary to Designate a Default Bean**



🔹 2. **Inject by Concrete Class Instead of Interface**

A close-up of a logo

AI-generated content may be incorrect.

🔹 3. **Use @ComponentScan Selectively**

Avoid registering multiple beans of the same type by refining your @ComponentScan.

A close-up of a computer screen

AI-generated content may be incorrect.

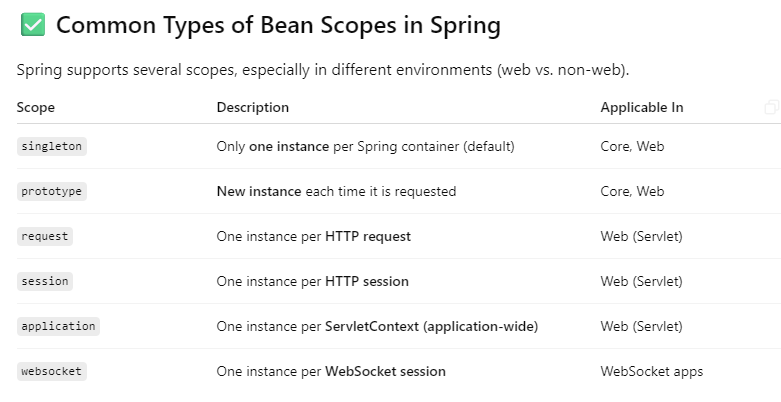
🔹 4. **Use Profiles to Load Only One Bean at a Time**

A screenshot of a computer program

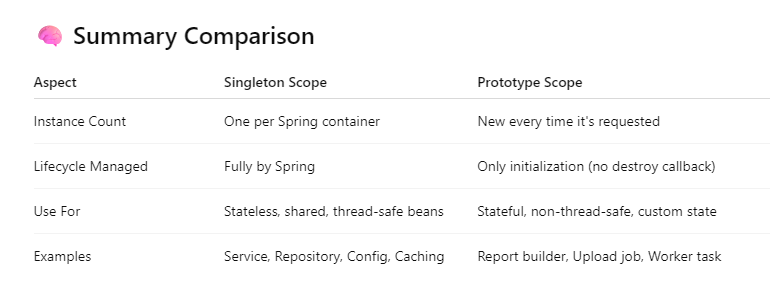
AI-generated content may be incorrect. A close-up of a website

AI-generated content may be incorrect.

1. **What is bean scope & Can you explain different type of bean scope ?**

****

1. **Can you provide a few real-time use cases for when to choose Singleton scope and Prototype scope ?**

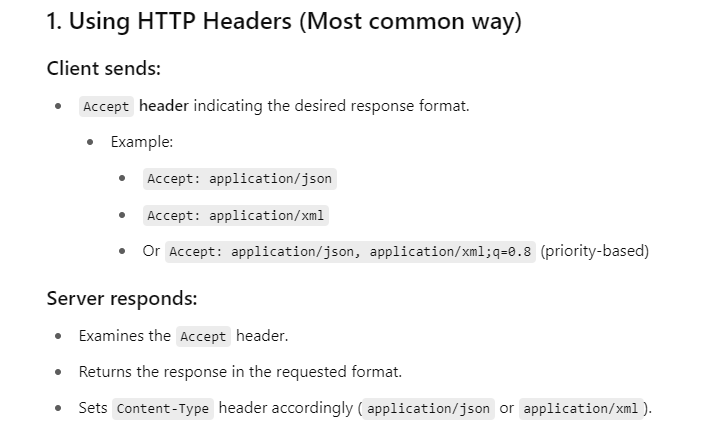
****

1. **How can we deserialize a JSON request payload into an object ?**

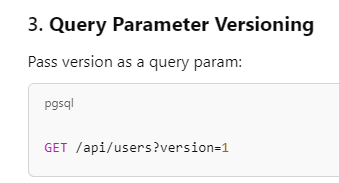
* **Jackson**

1. **How can we perform content negotiation (XML/JSON) in Rest endpoint ?**

A screenshot of a computer

AI-generated content may be incorrect.****

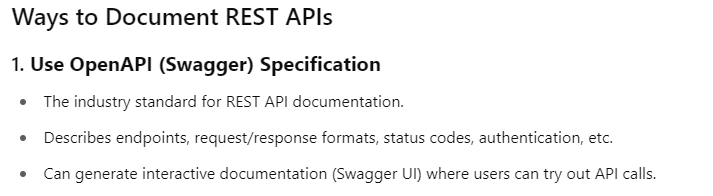
1. **How do you maintain the versioning for your REST API?**

**A screenshot of a computer

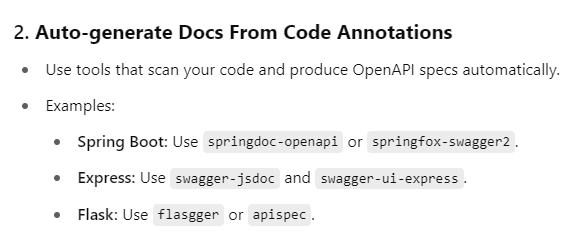
AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

1. **How will you document your rest API ?**

****

**A screenshot of a white background

AI-generated content may be incorrect.**

1. **How will you consume restful API ?**

****

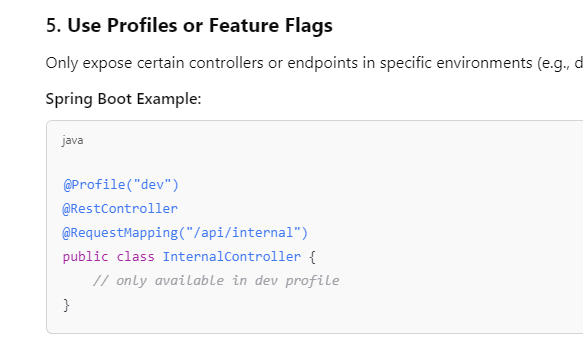
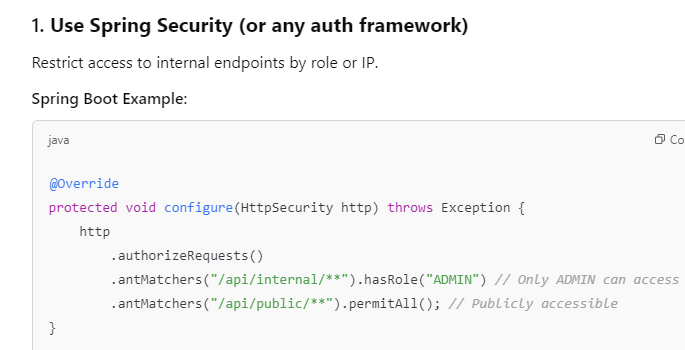
**A screenshot of a computer program

AI-generated content may be incorrect.**

**29 ) How can you hide certain REST endpoints to prevent them from being exposed externally?**

**A screenshot of a computer

AI-generated content may be incorrect.**

****

**30 ) How can you avoid defining handlers for multiple exceptions, or what is the best practice for handling exceptions ?**

* Instead of writing try-catch blocks or separate handlers in every controller , use a **centralized exception handler** to handle all exceptions in one place.

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A white text with black text

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**31 ) How will you validate or sanitise your input payload ?**

* + - 1. **Use Bean Validation**

Use annotations like @NotNull, @Email, @Size, etc., in your DTOs with javax.validation or jakarta.validation.



A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer error

AI-generated content may be incorrect.

**32) How can you populate validation error message to the end users ?**

* To **populate validation error messages to end users**, especially in REST APIs (like with Spring Boot), you need to:

1) **Use validation annotations** in your DTO.

2) **Catch validation errors** centrally.

3) **Return a structured, user-friendly error response**.

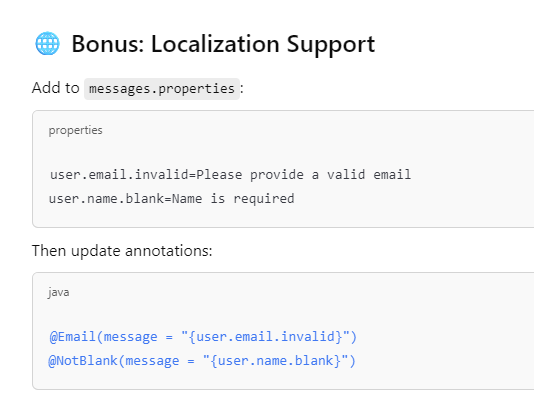
**🔹 1. Add Validation Annotations in DTO 2. Use @Valid in controller**

A screenshot of a computer code

AI-generated content may be incorrect.

**3 . Handle validation with @ControllerAdvice**

A screen shot of a computer code

AI-generated content may be incorrect.



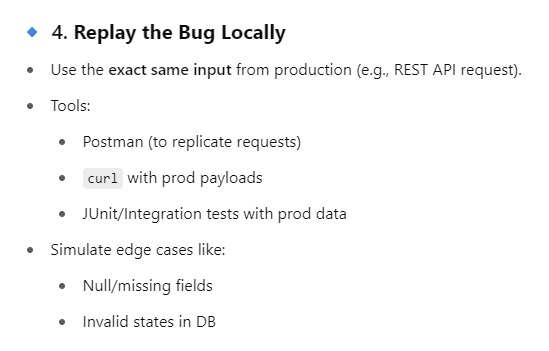
**33) Let’s say you find a bug in production environment and now you want to debug that scenario ,How can you do that from your local ?**

* **Debugging a bug found in the production environment from your local can be tricky.**

**A screenshot of a computer

AI-generated content may be incorrect.A white background with black text

AI-generated content may be incorrect.**

**A close-up of a message

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.**

**A white background with black text

AI-generated content may be incorrect.A close-up of a message

AI-generated content may be incorrect.A close-up of a text

AI-generated content may be incorrect.**

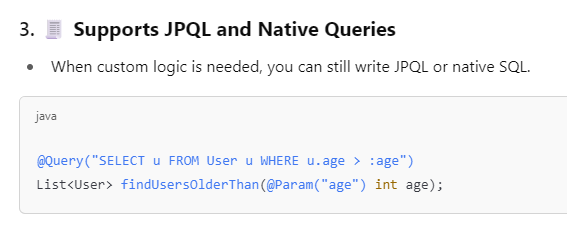
**34 ) What is the key benefit of using the Spring Data JPA ?**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a computer program

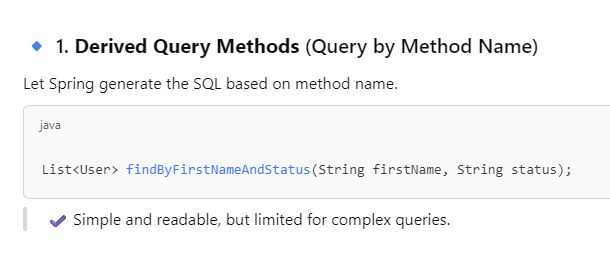
AI-generated content may be incorrect.**

**A white background with black text

AI-generated content may be incorrect.**

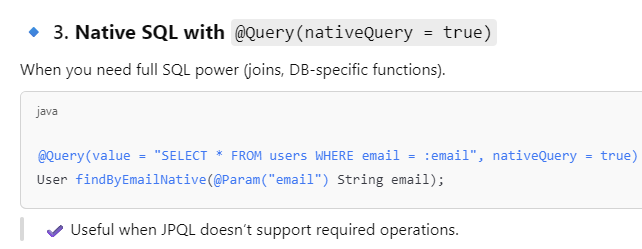
**35) What are the different ways to define custom queries in Spring Data JPA ?**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

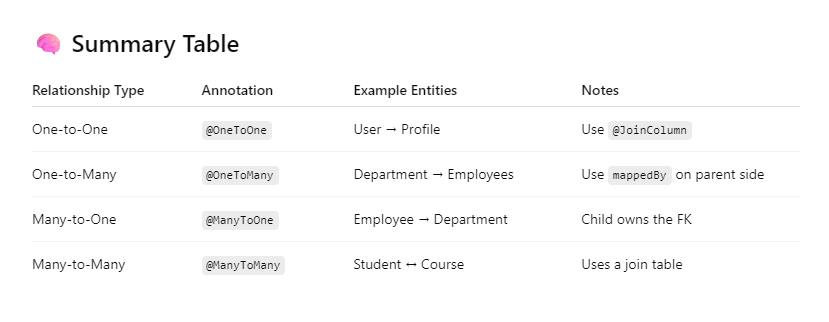
****

**A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

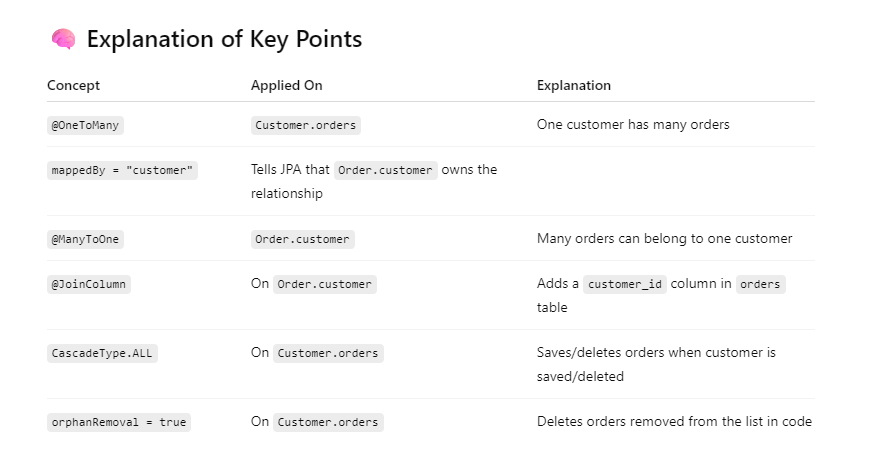
AI-generated content may be incorrect.**

**36) Relationship mapping ?**

****

**A screenshot of a computer code

AI-generated content may be incorrect.**

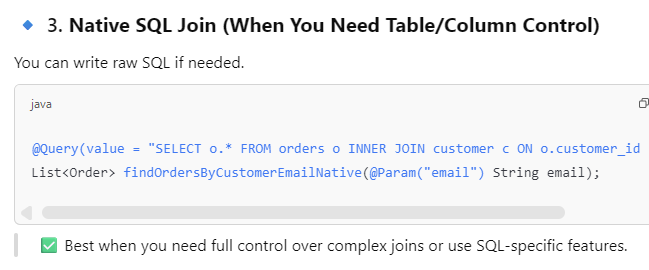
****

**37 ) Is this possible to execute Join query in Spring Data JPA ? If yes, how can you add some insights ?**

**A screenshot of a chat

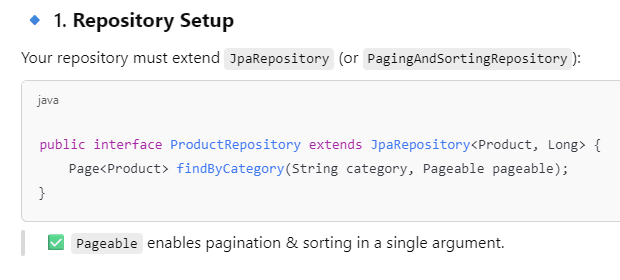
AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

****

**A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.38 )  How will you implement pagination & Sorting in Spring Data JPA ?**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a web page

AI-generated content may be incorrect.**

**39) What is transaction management and how it works ? explain**

**🡪 Transaction Management in Spring (or any enterprise app) ensures that a set of operations either all succeed or all fail as one atomic unit of work.**

**✅ How Transaction Management Works in Spring**

* Spring provides declarative transaction management using the @Transactional annotation.

****

**✅ All DB operations within this method are part of one transaction. If any exception occurs, everything is rolled back automatically.**

A screenshot of a computer

AI-generated content may be incorrect.A black text on a white background

AI-generated content may be incorrect.

A screenshot of a computer error message

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.A white box with black text

AI-generated content may be incorrect.

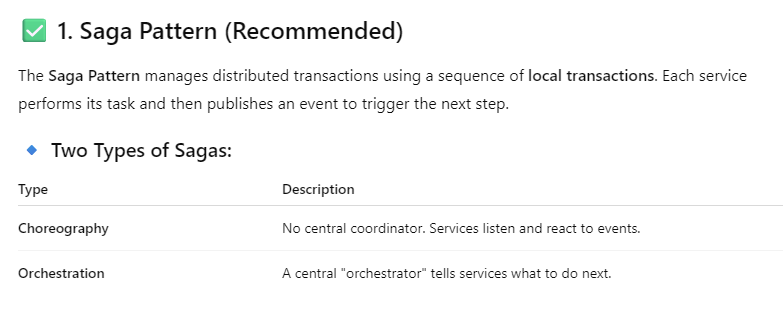
**40 ) Can we use transaction in private methods ?**

* **No, you cannot use @Transactional effectively on private methods in Spring.**

**41 ) How will you handle transaction in distributed microservice ?**

* **Handling transactions in distributed microservices is one of the most challenging aspects of microservices architecture.**
* **Since each microservice has its own database, traditional ACID transactions don’t work across services.**
* **Therefore, we use eventual consistency and patterns like**

A white background with black text

AI-generated content may be incorrect.****

**42) Explain orchestration and choreography pattern with example ?**

**🌀 1. Choreography-Based Saga (Event-Driven)**

**Flow:**

**Each service listens to events and publishes the next event.**

**No central control.**

**✅ Use Case: E-commerce Order Placement**

**Microservices involved:**

* 1. **Order Service**
  2. **Inventory Service**
  3. **Payment Service**

**🔸 Dependencies:**

* **Spring Boot**
* **Kafka (or RabbitMQ)**

A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.**A screenshot of a computer program

AI-generated content may be incorrect.**

A screen shot of a computer program

AI-generated content may be incorrect.

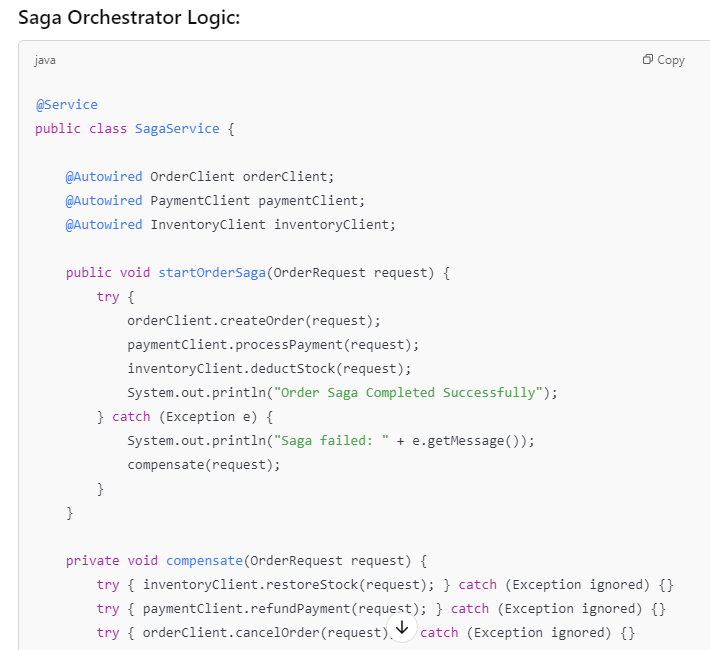
A screen shot of a computer program

AI-generated content may be incorrect.

**🧭 2. Orchestration-Based Saga**

**Flow:**

* A central **Orchestrator service** controls the steps and invokes each service.
* 💡 All services must expose APIs for compensate actions (cancel, release, refund).



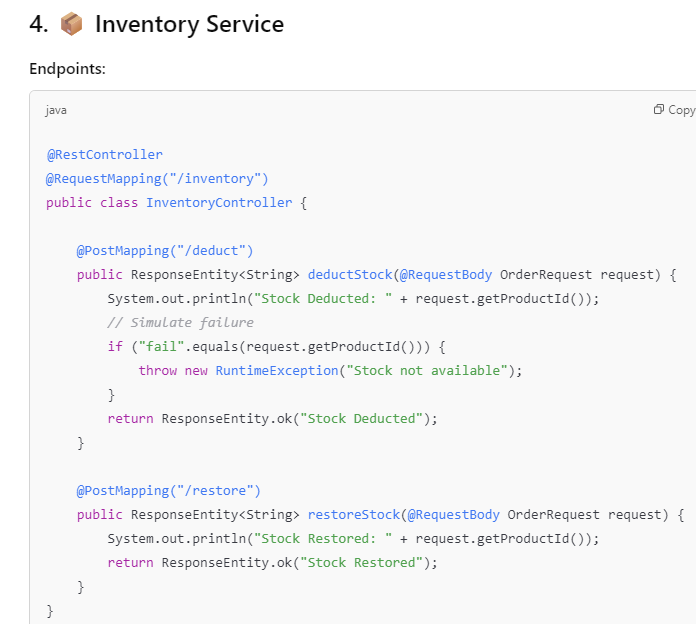
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.



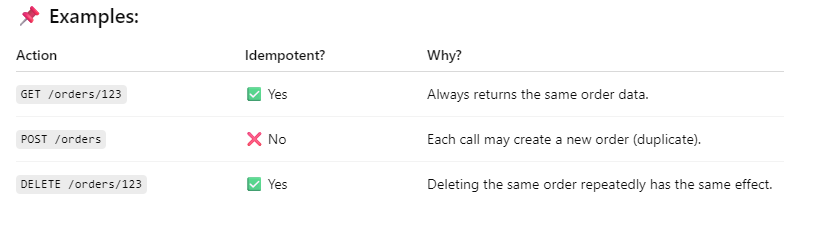
A screenshot of a computer program

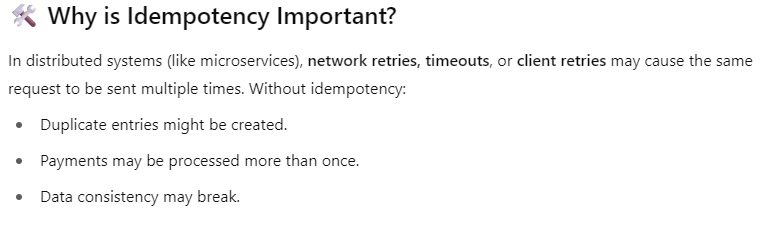
AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

**43 ) Idempotency Situation ?**

* Idempotency means that performing the same operation multiple times produces the same result as doing it once.



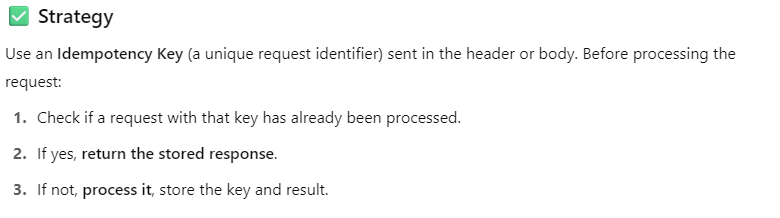


* Make non-idempotent operations (like POST) **idempotent** by handling duplicate requests gracefully.

**🧑‍💻 How to Implement Idempotency in Spring Boot 📦 Step-by-Step Code**

A screen shot of a computer code

AI-generated content may be incorrect.

****

**A white background with black text

AI-generated content may be incorrect.**

A close-up of a white background

AI-generated content may be incorrect.

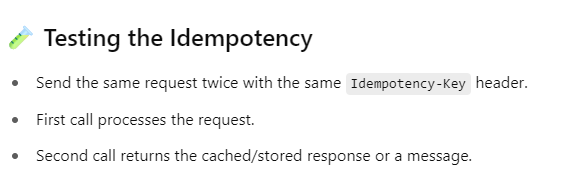
A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

* The **idempotency key** is **typically generated by the client (frontend/UI)** and sent with each request to the backend.

A screenshot of a computer

AI-generated content may be incorrect.

A white text on a black background

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

A screenshot of a computer code

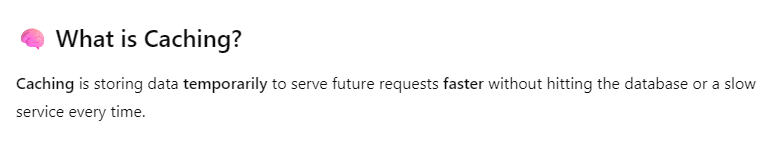
AI-generated content may be incorrect.

**44)** **Discuss Spring Boot’s approach to database migrations. How can one integrate tools like Flyway or Liquibase?**

* Spring Boot integrates seamlessly with database migration tools like Flyway and Liquibase, automating the process of database schema creation and updates.
* To use Flyway, add its starter dependency to your pom.xml or build.gradle.
* Spring Boot automatically detects Flyway migrations in **src/main/resources/db/migration** and applies them accordingly.
* Spring Boot automatically detects Flyway migrations in **src/main/resources/db-changelog.sql** and applies them accordingly.

**45) How do you implement caching in Spring Boot for improved performance?**

**A screenshot of a web browser

AI-generated content may be incorrect.**

**A white background with black text

AI-generated content may be incorrect.**

A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a white background

AI-generated content may be incorrect.

**A screenshot of a computer code

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

**46) What are the best practices for managing application properties and secrets in Spring Boot?**

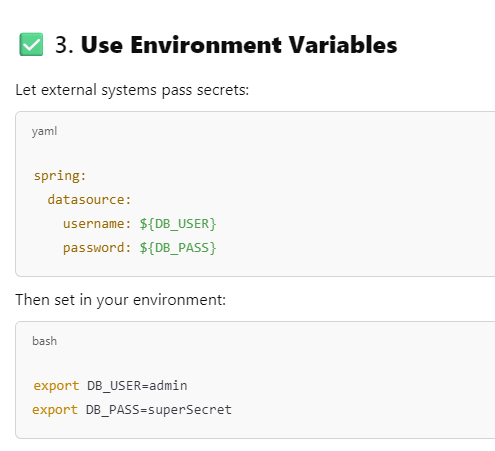
* Managing application properties and secrets in Spring Boot is crucial for security, maintainability, and environment-specific configuration.

🧩 1. **Organize application.properties or application.yml**

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer secret

AI-generated content may be incorrect.

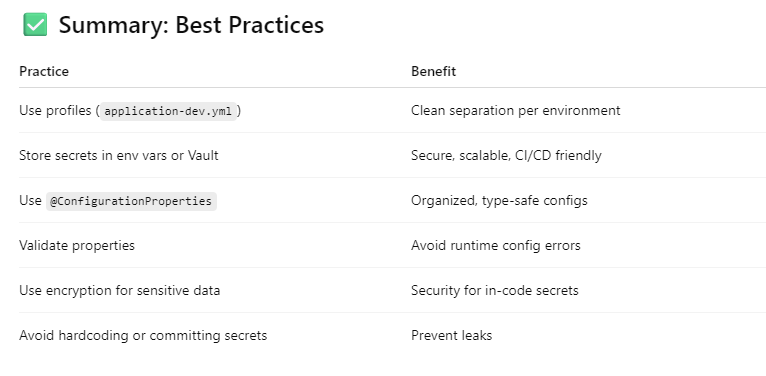


A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**47 ) What are the tools available for detecting memory leak ?**

**VisualVM**

* **VisualVM is an all-in-one Java troubleshooting tool that integrates several JDK command-line tools .**
* **It is lightweight performance and memory profiling capabilities.**
* **It’s included in the Oracle JDK download.**

**Key Features:**

* **Monitor application memory consumption in real-time.**
* **Analyze heap dumps to identify memory leaks.**
* **Track down memory leaks with its built-in heap walker.**

**Eclipse Memory Analyzer (MAT)**

* **Eclipse MAT is a specialized tool designed for analyzing heap dumps.**
* **It is particularly effective in identifying memory leaks and reducing memory consumption.**

**Key Features:**

* **Analyze large heap dumps.**
* **Automatically identify memory leak suspects.**
* **Provide detailed reports on memory consumption by objects.**

**Usage Example:**

**After obtaining a heap dump from a running application**

**(which can be triggered in the JVM on OutOfMemoryError),**

**MAT can be used to analyze this dump.**

**It provides a histogram of objects in memory, allowing developers to see which classes and objects are**

**consuming the most memory.**

**JProfiler**

* **JProfiler is a comprehensive profiling tool for Java with capabilities for both memory and performance profiling.**
* **It’s a commercial tool but is widely regarded for its user-friendly interface and detailed analysis.**

**Key Features:**

* **Real-time memory and CPU profiling.**
* **Advanced heap analysis and visualization.**
* **Ability to track every object in the heap and analyze memory consumption.**

**48 ) How do you implement a custom health check in Spring Boot Actuator?**

* To implement a custom health check in Spring Boot Actuator, you need to create a bean that implements the **HealthIndicator** interface.
* Spring Boot will automatically include your custom health check in the /actuator/health endpoint.

A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**49) How would you handle a situation where a Spring Boot application runs out of memory?**

🡪 When a Spring Boot application runs out of memory (OOM), it's often due to memory leaks, inefficient data structures, or misconfigured JVM settings.

A screenshot of a computer error message

AI-generated content may be incorrect.

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

**A screenshot of a computer code

AI-generated content may be incorrect.**

**50) How would you design a Spring Boot application to handle millions of concurrent users?**

🡪 Designing a Spring Boot application to handle millions of concurrent users requires careful planning across architecture, scalability, performance

A screenshot of a computer

AI-generated content may be incorrect.optimization, and fault tolerance.

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A white background with black text

AI-generated content may be incorrect.

A white sign with black text

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

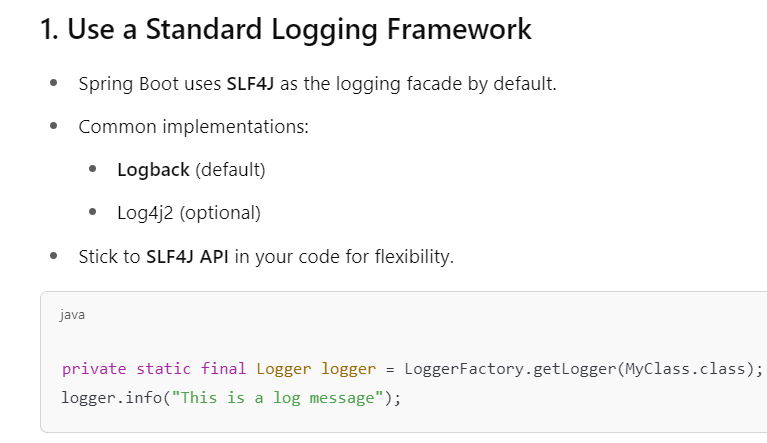
A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**51) What are the best practices for logging in Spring Boot?**

**A screenshot of a computer error

AI-generated content may be incorrect.**

**A screenshot of a computer error

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

**52) What are the best practices for securing a Spring Boot application?**

🡪 Securing a Spring Boot application involves multiple layers—from authentication and authorization to protecting against common vulnerabilities and securing infrastructure.

A white background with black text

AI-generated content may be incorrect.A white background with black text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A close-up of a computer screen

AI-generated content may be incorrect.

A screenshot of a white background

AI-generated content may be incorrect.

**53) How do you ensure code quality in a Spring Boot project?**

A white background with black text

AI-generated content may be incorrect.A white text on a white background

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A white background with black text

AI-generated content may be incorrect.

A white text with black text

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.

A black text on a white background

AI-generated content may be incorrect.A close-up of a code

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

**55) How do you implement OAuth2 authentication in Spring Boot?**

**✅ Steps to Implement OAuth2 Client Authentication in Spring Boot**

**A screenshot of a computer code

AI-generated content may be incorrect.A white background with black text

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

A black text on a white background

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.

**✅ Securing REST APIs (OAuth2 Resource Server)**

**A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.** **A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer code

AI-generated content may be incorrect.**

**56 ) What is the difference between Basic Authentication and JWT in Spring Boot?**

* The difference between Basic Authentication and JWT (JSON Web Token) in Spring Boot lies in how authentication information is transmitted, stored, and managed.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**A white background with black text

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

**57) How do you implement role-based access control (RBAC) in Spring Boot?**

**A screenshot of a computer program

AI-generated content may be incorrect.58) How do you optimize database queries in Spring Boot using JPA or Hibernate?**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer error

AI-generated content may be incorrect.** **A screenshot of a computer

AI-generated content may be incorrect.**

**58) How would you handle a situation where a Spring Boot application fails to start due to a circular dependency?**

**A black and white text on a white background

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**