**Lab Assignment: Implementing Microservice Communication (Part 1)**

**Objective:**

In this lab, you will build three microservices: **Product Service**, **Order Service**, and **Inventory Service** using Spring Boot. These microservices will interact with MongoDB & MySQL databases and expose RESTful APIs. You will also write integration tests to ensure everything is functioning properly.

By completing this lab, you will gain hands-on experience in building, configuring, and testing microservices with Spring Boot and other relevant tools.

**Instructions:**

1. **Follow** [**the tutorial**](https://programmingtechie.com/articles/spring-boot-microservices-tutorial) to build the three microservices: **Product Service**, **Order Service**, and **Inventory Service**.
2. **Set up each service** with the appropriate configurations, entities, and endpoints.
3. **Test each service** using Postman and integration tests with TestContainers.
4. **Answer the questions** at the end of the lab to reflect on the concepts covered.

**Post-Tutorial Questions:**

1. Why is it important to create separate databases for each microservice (e.g., product\_service, order\_service, inventory\_service)?
2. What role does Flyway play in managing the database schema, and how does it ensure consistency across environments?
3. How does Spring Data JPA simplify working with databases in each of the microservices?
4. In the InventoryService, why did we use the @Transactional(readOnly = true) annotation, and what is its significance?
5. In a microservices architecture, what are some challenges when ensuring communication between the Product, Order, and Inventory Services?
6. What are the advantages of using TestContainers for integration testing with MySQL in this lab?

**Submission:**

* Submit your code through your preferred version control system (e.g., GitHub) and include the link to your GitHub repo.
* Include any Postman collections, test results, or logs generated during testing.
* Please submit the answers to questions in a readme named READMELAB4.md.

**Additional Notes:**

* Each microservice should be independently testable.
* Pay attention to how each microservice communicates with its own database.
* Make sure to handle exceptions and errors properly to ensure production-grade microservices.

Good luck, and happy coding!