Task

**Scenario 1:** You’re working on an e-commerce application that has two services: OrderService and InventoryService. When a customer places an order, OrderService should check with InventoryService to confirm that the items are in stock.

* Questions:

1. How would you make OrderService communicate with InventoryService to check the stock?
2. Would you prefer direct HTTP calls or use a message broker? Why?
3. What issues could arise if InventoryService becomes temporarily unavailable?

**Scenario 2:** Your application has multiple services (UserService, OrderService, PaymentService) that need to interact. You want each service to automatically find and connect with others.

* Questions:

1. What is service discovery, and why is it important in a microservices architecture?
2. How would you implement service discovery in this application?
3. If you use a service discovery tool (like Consul or Eureka), how will UserService find OrderService?

**Scenario 3:** You have a payment processing service (PaymentService) that is used by the OrderService when customers check out. If PaymentService is down, the OrderService should not crash.

* Questions:

1. How would you handle this dependency to make OrderService resilient?
2. What is a fallback mechanism, and how could it be used here?

Have you heard of a circuit breaker pattern? Can you explain how it might help in this situation?

**Scenario 4**: You have multiple instances of the ProductService running to handle high traffic. When a user requests product data, the request should go to an available instance.

* Questions:

1. What is load balancing, and why is it important in a microservices architecture?
2. How would you set up load balancing for the ProductService?
3. What would happen if one instance of ProductService fails?

**Scenario 5:**Each microservice in your application has its own database, but you notice that some services still need data from other services, like Order Service needing customer information from the Customer Service.

1. How would you avoid direct database access between services to maintain data isolation?
2. Would you consider data replication or a separate data store for frequently accessed data? Why?
3. What pattern would you use to handle the need for data from another service without breaking microservices principles?