Task: Develop a Microservice-based Order Management System

Objective:

You are required to build a simple order management system that handles order creation, inventory management, and order fulfillment using Spring Boot. The system should be composed of multiple microservices, communicate asynchronously using RabbitMQ, and utilize MySQL for data persistence. The entire application should be containerized using Docker.

Requirements:

1. Microservices:

- **Order Service:**
    - Expose an API to create a new order.
    - Communicate with the Inventory Service to check the availability of items before placing the order.
    - Save the order details in a MySQL database.
    - Publish an event to RabbitMQ when an order is successfully created.
- **Inventory Service:**
    - Expose an API to check item availability.
    - Reduce the inventory count when an order is placed.
    - Communicate back with the Order Service if the item is out of stock.
    - Save the inventory details in a MySQL database.
- **Shipping Service:**
    - Listen for order creation events from RabbitMQ.
    - Process the order for shipment.
    - Update the order status to "Shipped" in the Order Service through a REST API call.

2. Technology Stack:

- **Spring Boot:** For developing all the microservices.
- **MySQL:** For persisting order and inventory data.
- **RabbitMQ:** For asynchronous communication between services.
- **Docker:** Containerize each microservice and use Docker Compose to manage them.

3. Endpoints:

- **Order Service:**

    - POST /orders: Create a new order.
    - GET /orders/{orderId}: Get details of a specific order.
- **Inventory Service:**
    - GET /inventory/{itemId}: Check availability of an item.
    - POST /inventory/reduce: Reduce inventory after an order is placed.
- **Shipping Service:**
    - Listen for order creation messages and mark orders as shipped.

4. Additional Requirements:

- **Docker Compose File:** Create a docker-compose.yml file to orchestrate the services, MySQL database, and RabbitMQ.
- **Documentation:** Provide a README file with instructions on how to run the services using Docker.
- **Unit Testing:** Write unit tests for the key components using JUnit.

5. Bonus:

- Implement a simple API Gateway using Spring Cloud Gateway to route requests to the appropriate microservice.
- Implement basic authentication and authorization using Spring Security.

Delivery:

- Please provide a GitHub repository with the complete source code, Docker files, and documentation.
- Ensure that the system can be run locally using Docker Compose with minimal setup.

Timeline:

- **Duration:** 2 weeks
- Kindly update me periodically with your progress (e.g., after the first week).

Contact:

- If you have any questions or need clarification, feel free to reach out to me on Telegram: **@aminmardani79**.