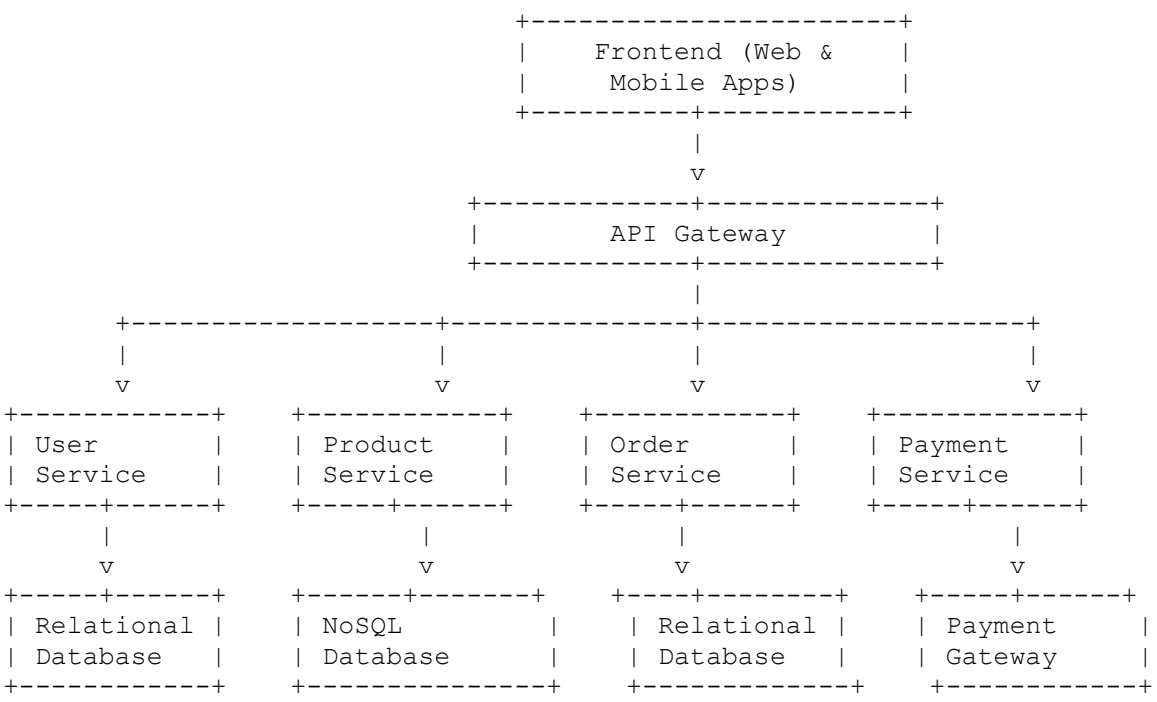
Project Title: Simulating E-Commerce System Architecture using WireMock

Objective: The objective of this project is to simulate the architecture of an e-commerce system using WireMock. You will create mock endpoints to replicate the behavior of various services within the system, allowing for isolated and controlled testing of APIs without relying on actual service dependencies.

System Architecture

Detailed Architecture Diagram:



The e-commerce system consists of the following services:

1. **User Service:** Manages user accounts, authentication, and user profiles.
2. **Product Service:** Manages product listings, product details, and inventory.
3. **Order Service:** Handles order creation, order details, and order status.
4. **Payment Service:** Manages payment processing, payment status, and transaction history.

Project Requirements

1. **Environment Setup:**
    - Install wiremock
2. **WireMock Configuration:**
    - Set up WireMock as part of your test suite.

    - Configure WireMock to run on port 8080.
3. **Directory Structure:**
    - Create a directory structure for WireMock mappings and files.

```
css
Copy code
src/main/resources
└── wiremock
    ├── mappings
    └── __files
```

4. **Mock Endpoint Creation:**
    - Create mock endpoints for each service in the e-commerce system.
    - Define request and response mappings for each endpoint.
5. **Response Files:**
    - Create response files to be used by the mock endpoints.
6. **Test Case Design:**
    - Design test cases to validate the behavior of the simulated system.
    - Include scenarios for successful interactions, error handling, and edge cases.
7. **Postman Collection:**
    - Create a Postman collection to test the mock endpoints.
    - Apply Postman best practices such as environment variables, scripting, and proper organization.

Deliverables:

- Complete source code of the project.
- Configuration files for WireMock.
- Postman collection and environment.
- Test cases and results.
- Project documentation summarizing your approach and findings.

Evaluation Criteria:

- Completeness and correctness of mock endpoints.
- Effective use of WireMock features.
- Organization and structure of the project.
- Quality and comprehensiveness of the test cases.
- Proper usage of Postman best practices.
- Clarity of the project documentation.

Bonus:

- Implement advanced WireMock features such as stateful behavior, delays, and fault injection.
- Simulate scenarios like service unavailability, timeouts, and rate limiting.