MediaMarktSaturn
Technology

Coding Challenge: Marketplace

The MediaMarktSaturn Marketplace allows online sellers to offer their products on our MediaMarkt and Saturn touchpoints (e.g. webshop / mobile app). The aim is to meet customers' needs through a diverse product portfolio.

The following scenario is meant to evaluate your capabilities, experience and seniority in programming modern web applications. Specifically, we want you to build a simplified marketplace ecommerce system for ordering and invoicing which covers the life cycle of an order and manages its state transitions. The system should include two microservices which communicate asynchronously with each other where it makes sense. The result should be presented by you in person in a technical interview.
An order has 5 regular statuses: Created, Accepted, Rejected, Shipping in progress and shipped. The order creation should be triggered by a REST endpoint. To simplify the data model we assume that an order can only reference one product. In addition to the status the order should have the properties order id, price, quantity, a product id, a customer id and seller id. Since products and customers are out of scope the ids can be any id. To allow sellers to manage orders the service should provide endpoints for listing orders, checking the details of an order, and updating an order.

The invoice services should allow sellers to upload invoices as PDFs for their orders. When an order has an invoice, and the order is in status Shipped the invoice should be sent out. In our simplified application it is enough to store a timestamp on the invoice when it was sent out. In addition to the timestamp the invoice should have the properties invoice id and order id.

Requirements

- NodeJS (>= 16) with TypeScript (>= 4.5)
- NoSQL database (e.g. MongoDB)
- Microservice Architecture
- RESTful APIs
- Asynchronous communication via message broker/queue (e.g. Kafka, RabbitMQ, Pub/Sub)
- Local development setup
- Basic testing

Bonus Tasks *(not required)*

- Create a production Dockerfile for each of the microservices
- Create a CI pipeline for the CI tool of your choice and add the CI file to the project
- Two types of users (sellers and customers) with different access rules for each service
- Every important aspect of the application shall be tested

Evaluation Criteria

- Requirements fulfilled
- Code structure and readability of the code
- Software architecture
- Used frameworks and tools
- Testing strategy

Final Remarks

- Most important for us is to understand how you tackle complex problems. Therefore, please finish this challenge as far as possible under the given technological requirements. A correct and understandable partial implementation is more important than a complete one.
- The assignment will be presented in a live demo as part of the personal interview.
- Please make sure the assignment works offline on your machine.

Happy Coding!