Microservices Certification Training

Certification Project

edureka!

Problem Statement

Background:

A popular library ("**Amazing Books**") in the city is planning to launch a portal for expanding their reach. They hire you to help them, go online. They want to proceed with incremental but fast-paced releases to attract more readers. They have a few major functions which should be independently developed, tested, and released from other functions to enable fast-paced releases. Apart from this, they expect the user base to grow soon and thus must be highly scalable without compromising on availability, resiliency, and security.

The portal would need to support different types of devices eg. web, mobile, tablet, and thus would require RESTful APIs which you need to build to manage issuing of books to the readers. Development of GUI (Graphical User Interface) would be taken up by another team and the delivery of the books will be handled by another vendor.

So, there are three major functions that need to be implemented:

1. APIs to fetch/add/edit/delete books (H2 Database).
2. APIs to fetch/add/cancel issuing of the books to the customers (H2 Database).
3. Customers to be managed by OAuth2 server(in-memory)

Apart from these functional requirements, you would also have to take care of non-functional requirements:

1. It should be easy to scale.
2. It must be highly secured.
3. It should be resilient.
4. It should be performant.
5. It should support quick releases.

Considering the above-mentioned requirements (functional + non-functional), Microservices Architecture seems fit for this. Before the implementation do provide High Level Design.

Below is a list of important functions(microservices) and the model which you may consider for the implementation:

1. Book(bookms): fetch/add/edit/delete books
   Model: isbn, title, publishedDate, totalCopies, issuedCopies, author
   Assumption: A book will have only one author
2. Issue books to customers(issuerms)
   - Fetch(REST call) Book Details eg bookId and availableCopies(totalCopies - issuedCopies)
   - Issue the book to the customer, if the book is available
   - Update issuedCopies in bookms
   - To keep things simple, these operations may not necessarily be atomic
   Model: isbn, custId, noOfCopies
   Note: A book(isbn) can be assigned to more than one customer
3. A service for dynamic service discovery

4. A service for handling cross-cutting concerns
5. A service for managing security
   - Have one client application registered(in-memory)
   - Add customers as in-memory users

Recommendations:

1) Spring boot 2.5.x version or higher
2) Java 11
3) Use In-Memory(H2) Database wherever applicable
4) Use any IDE of your choice
5) Use JSON for exchanging data through RESTful APIs
6) For security, use OAuth2. Add customers to the OAuth2 server(in-memory)
7) Swagger Documentation
8) Dockerize your services
9) Focus should be on implementing several design patterns relevant in Microservices
10) Keep Data Model simple eg. A book may have only one author
11) Call out assumptions that you may make in implementation in terms of domain
12) Use OAuth2 server to manage users(in-memory)
13) Use Prometheus and Grafana to monitor your application

Deliverables:

1) High Level Design
2) Code of all services (five)
3) API sample request/response
4) Highlight any improvements that you may have incorporated

As a Microservices Developer, you must take into consideration the above-mentioned requirements and give appropriate solution.