Assignment 1: Reactive Programming - Microservices

Purpose

The goal of this assignment is to put into practice the reactive programming techniques we discussed and demonstrated in class and to combine this with what you learned about microservices in last semester's Web Services and Distributed Computing course.

Beginning with the provided Student Starter and the Courses-Service code-along done in class, you will complete the **Enrollment portion of a Registration System**.

In the end, it will consist of three microservices: students-service (provided), courses-service (code-along), and enrollments-service (started in class and completed by you). The databases and docker/docker-compose are already configured in the Student Starter.

This is an individual assignment.

Due

This assignment is due Tuesday, September 3, 2024. It will be worked on in class. Any work that cannot be completed in class will need to be completed as homework. Your grade will be based on Peer Evaluation. **If you miss class without a college-approved absence, you will receive a grade of 0 for this assignment**.

Be sure to check the evaluation section before you begin to understand how you will be evaluated for this assignment.

Materials

- enrollment-system-ms-reactive – student starrter.zip (on Moodle)
- You will need IntelliJ Ultimate

- You will need Docker/Docker-Compose
- You will need GitBash
- You will need the lecture notes from the Introduction to Reactive Systems and Introduction to Reactive Spring

Required Tasks

See the rubric for the specifics on how everything will be graded. The list below is a list of tasks.

Enrollments-Service

1. Complete the development of the Enrollments-Service. It should support the following APIs:
    a. Post (done as a code-along)
    b. PutEnrollmentByEnrollmentId
    c. GetAll
    d. GetEnrollmentByEnrollmentId
    e. DeleteEnrollmentByEnrollmentId

2. Implement a Global Controller Exception Handler. Ensure that at least two exceptions are thrown in the code and handled by the GlobalControllerExceptionHandler.

3. Implement Domain Client Exception Handling. Ensure that at least two exceptions are thrown by the courses-service and handled in the CourseClient.

Notes:
addEnrollment and updateEnrollmentByEnrollmentId must send a request to both the Students-Service and the Courses-Service to get new/updated information based on the provided StudentId and CourseId in the EnrollmentRequestModel.

For add, a new document must be added to the database.

For update, no new document must be added to the database.

Courses-Service

1. Complete the development of the Courses-Service (if you missed part of the code-along). It should support the following APIs:
    a. Post
    b. PutCourseByCourseId
    c. GetAll
    d. GetCourseByCourseId
    e. DeleteCourseByCourseId

2. Implement Global Controller Exception Handling. Ensure that at least one NotFoundException and one InvalidInputException are thrown by your code.

Submission Requirements & Peer Evaluation Procedures

a) Submit your code on Moodle ON TIME.
b) Peer Evaluation – this assignment will be demonstrated to one of your peers who will fill out a checklist supplied by the teacher (that matches the list in the Evaluation section below). Be prepared to demonstrate (i.e. make sure your Postman queries are already set up as you will only have about 5 minutes to demonstrate) the following:

1) Demonstrate that all 9 required docker containers come up with **docker-compose up**. Then show their healthy status using **docker ps**.

2) **For each endpoint of the courses-service**, demonstrate that it functions as expected using Postman (or curl/jq) and the database viewing tool (pgadmin):
    a. Post
    b. PutCourseByCourseId
    c. GetAll
    d. GetCourseByCourseId
    e. DeleteCourseByCourseId

3) **For courses-service,** demonstrate that GlobalControllerExceptionHandler has been implemented by sending two API requests that will trigger:
    a. a NotFoundException and
    b. an InvalidInputException.

4) **For each endpoint of the enrollments-service**, demonstrate that it functions as expected using Postman (or curl/jq) and the database viewing tools (pgadmin, phpmyadmin, and mongo-express).
    a. For Post, show that a new enrollment is created. Show that it contains the correct data from the StudentsDB and from the CoursesDB by opening up all three database viewing tools.
    b. For update, show that a new enrollment is NOT created and that the existing enrollment is updated. Be sure to include a test where the courseId is changed and another where the studentId is changed. Open up all three database viewing tools to demonstrate changes to the database.
    c. For getAll/getEnrollmentByEnrollmentId, show that the correct data is retrieved from the database (show the mongo-express viewing tool).
    d. For deleteEnrollmentByEnrollmentId, show that the correct enrollment has been deleted from the database (show the mongo-express viewing tool).

5) **In the Enrollments-Service**, demonstrate that Global Controller Exception Handling and Domain Client Exception Handling have been implemented. You will do this by showing that at least two exceptions have been handled in the code in the following manner:
    a. Make a PUT request where the CourseId is not found in the Courses-Service. Demonstrate that a NotFoundException is thrown.
    b. Make a getEnrollmentByEnrollmentId where the EnrollmentId is not a UUID. Demonstrate that an InvalidInputException is thrown.

6) **Peer Feedback** – you will receive a score on the quality and accuracy of the feedback provided to your peer in the peer evaluation activity.

Evaluation

This assignment is worth 4% of your grade this semester. The assignment will be graded according to the criteria in the table below. Be sure to read it attentively.

Task/Requirement	Max Points
1. Nine required docker containers	1
2. Courses-service endpoints	
a. Post	2
b. PutCourseByCourseId	2
c. GetAll	1
d. GetCourseByCourseId	1
e. DeleteCourseByCourseId	1
3. Courses-service exception handling	
a. NotFoundException	1
b. InvalidInputException	1
4. Enrollments-service endpoints	
a. Post	1
b. PutEnrollmentByEnrollmentId	4
c. GetAll	1
d. GetEnrollmentByEnrollmentId	2
e. DeleteEnrollmentByEnrollmentId	1
5. Enrollments-service exception handling	
a. Put request with NotFoundException on CourseId	2
b. GetEnrollmentByEnrollmentId with InvalidInputException on EnrollmentId	2
6. Peer Feedback	2
Total	25