ALGONQUIN COLLEGE | Online

Assignment #3 – Development Environment - components and setup [15%]

This assignment relates to the following Course Learning Requirements:

CLR 2: Install and use enterprise programming and deployment tools.
CLR 3: Implement Web Server integration with enterprise applications.
CLR 6: Implement and Integrate various Java based technologies used in the enterprise environment.

Objective of this Assignment:

Ensure that you are able to implement the knowledge obtained from module content into real-life code.

With that, you should be able to demonstrate the following:

- Working environment
- Working services
- Working intercommunication between services

Pre-Assignment Instructions:

1. To prepare you for this assignment, read the module 8 content and follow the embedded learning activities.
2. For this assignment it is required to have installed Ubuntu VM. This installation was discussed in module 8, which subsequently guided you to the setup of microK8s cluster.
   a. *NOTE: If you prefer to setup your K8s cluster in Windows and choose minikube instead of microK8s – that's not a problem as well.*
3. The other required software for this assignment is the installation and setup of an IDE of your choice
   a. IntelliJ IDEA, Eclipse or Netbeans sitting inside your VM.
4. As a final pre-requisite to this assignment, a database installation must be inside your local cluster.
   a. For simplicity, you can use MySQL v.8+ (https://dev.mysql.com/downloads/).
   b. For instructions on installing the DB inside your K8s cluster, please read the following: https://kubernetes.io/docs/tasks/run-application/run-single-instance-stateful-application/.
      i. *NOTE: Pay attention – the version of the MySQL cluster, used in the example above is 5.6. If you're OK with that you may follow copy-paste pattern, otherwise you need to change the version to whatever you want from DockerHub. Another thing to pay attention to is the password in the deployment file.*

Assignment description:

Remembering the Basic Functionalities from Assignment #1

In the first assignment, you started development on a backend for a Twitter-like application. Please remember its basic functionality:

1. There should be at least 2 roles – Producer and Subscriber;
2. Producer role is the same as Subscriber but has some extra capabilities – it can produce messages, which Subscribers get;
3. User may have both roles at the same time;
4. Users having a Subscriber role may subscribe to as many Producers as they want to;
5. All messages stored in the database, can be easily searched based on at least the criteria:
    a. User (Producer) ID – means who wrote it;
    b. Message content

Understanding the general concept for this application from Assignment #2

To recap Assignment #2, as well you were provided with the following as a general concept for your application.

1. There's a back-end application which has an exposed API allowing users to be authenticated and authorized (use data model from module 5) – UserManagementService;
2. User must be logged in and have obtained a token. For simplicity, imagine the token is a simple ID, like number, UUID, hash, something else, producing by the UserManagementService at the first authentication request and returned to the user for further authorization purposes.
3. After the user has been authenticated and obtained a token, he/she is able to use other APIs, related to publishing and obtaining messages; in case of unauthenticated access user should be simply redirected to UserManagementService to obtain a token; if user has a token but his/her rights are insufficient to perform a request – there should be appropriate response with HTTP status 401;
4. Using other APIs, the user is able to publish messages and/or receive messages published by other users.

Assignment #3 Overview

As you embark on your third consecutive assignment, at this point in your development, you should have finished all preparation steps in the 2 previous assignments. This preparation has led you to this assignment, in which you are required to convert your previous work into a working code according to the data model and UMLs you've previously created.

For simplicity purposes, consider the following:

1. All users trying to achieve an API for now need to provide a token correlated to any chosen user from UserManagementService; your service
2. This triggers API,
3. Therefore, you need to authorize any incoming request by making a blocking API call to UserManagementService to obtain permission for this chosen user by comparing it with the issue's tokens.
4. For now, use UUID as a token.
    a. That UUID should be generated for any user, requesting UserManagementService with username and password in case that username and password are fit to any record inside the UserManagementService's database.

Note: This is also a way to implement tokens for the application created in Assignment #2

5. As a package name for your application, please use the following pattern: org.ac.cst8277.{Last Name}.{Fisrt Name}
6. You should use Spring Boot Framework for your services with Netty as an application server (use the following dependency in your pom.xml file):

```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-webflux</artifactId>
</dependency>
```

Assignment Tasks:

Now that you have prepared all required content for your application in Assignments #1 and #2, you will now begin implementation of the following.

What you should do:

- Having a data model from assignment # 1 and UML diagrams from assignment # 2 you need to implement your services up to a working condition.
- As a result of this assignment you should get working and tested code providing basic Twitter functionality.

Submission Guidelines:
You need to submit an archive with name pattern {Course #}_{Section #}_{Last name}_{First name}.zip containing your project source code and short video file demonstrating the following:

1. How you build all services (either from terminal window or from your IDE) (5 points)
2. How you start all services and their status after start, including host and port they start listen to (5 points)
3. How you issue your requests: (5 points)
   I. Get all users
   II. Get all roles
   III. Add user
   IV. Get all messages
   V. Get messages produces by specific producer
   VI. Get messages for specific subscriber

Please pay attention your video should be quite short and not exceed the limit of 100Mb per file. It is not required to record yourself, your voice with comments, or anything else, unrelated to the working application. In my case it was enough to record a terminal window and an application I used to issue HTTP requests (Postman), so you may do the same.

Look at the rubric below to see the breakdown of the points.

Assignment # 3 Grading Rubric (15%)

Criteria	Code is fully functional using API calls and token 80-100%	The code is partially functional but lacks some requirements 50-79%	Code is not functional, but a basic understanding of the concepts is demonstrated <50%	Points
Build of services	The services are built properly and efficiently Clearly demonstrated in the video submission	Some of the services are built properly and efficiently Clearly demonstrated in the video submission	The services are not built properly, but the learner has demonstrated an attempt in doing so. No apparent demonstration in the video submission	/5
Comments				
Demonstration of Startup and status of built services	The services can clearly be seen as functioning including their status and the port number that they are running on. Clearly demonstrated in the video submission	Some of the services can clearly be seen as functioning including their status and the port number that they are running on. Clearly demonstrated in the video submission	The services cannot clearly be seen as functioning or do not function and fail to include their status and the port number that they are running on. No apparent demonstration in the video submission	/5
Comments				
Demonstration of issuing requests	The submission clearly demonstrates the listed requests being issued and returning the appropriate responses	The submission clearly demonstrates some of the listed requests being issued and returning the appropriate responses	The submission does not clearly demonstrate the listed requests being issued or returning the appropriate responses No apparent demonstration in	/5

Criteria	Code is fully functional using API calls and token 80-100%	The code is partially functional but lacks some requirements 50-79%	Code is not functional, but a basic understanding of the concepts is demonstrated <50%	Points
	Clearly demonstrated in the video submission	Clearly demonstrated in the video submission	the video submission	
Comments				
Total Points				/15