SWE645 – Assignment 4/Group Project

This assignment involves creating a Student Survey Data Broker that allows to publish and consume student survey records at scale. This entails setting up a Kafka cluster, creating a Kafka topic, developing producer/consumer applications to publish and read messages to/from the topic, and integrating this solution with the last homework. Here are specific details:

Student Survey Data Broker
The student data broker should be implemented using Strimzi Kafka operator to setup a Kafka cluster with at least three Kafka brokers and three ZooKeepers on a Kubernetes cluster. It's okay to use one node Kubernetes cluster, although you are more than welcome to use distributed worker nodes. The data broker should support message ordering (first-in-first-out) on a per partition basis. The broker deployed on the Kubernetes cluster should be accessible using either the nodeport, load balancer, or ingress. The broker must use persistent storage to save student survey records. The persistent storage can be implemented using persistent volume claim and storage class concepts of Kubernetes. If you are using AWS, you can use aws-ebs-csi-driver to provision a storage class and then create persistent volumes dynamically at the deployment of the broker based on persistent volume claim.

Message Topic
Create a Kafka topic called survey-data-topic (or any name of your choice) with at least three partitions and a configurable replication factor for the partitions. You can use a replication factor of 1 (one) for the partitions in this project.

Producer/Consumer
Write Java-based Producer and Consumer applications to publish and consume the student surveys to/from the survey-data-topic. Integrate your producer application with the last homework so that when you fill out and submit the completed survey form, the survey data gets published to the topic in Kafka broker as opposed to the table in the database that you implanted in your last homework.

Optional Extra Credit – Kafka Security
Incorporate authentication and authorization so that only authenticated and authorized producers/consumers can publish and consume data from the kafka topic. Implement SSL/TLS authentication using your own certificate authority. You will also need to implement some aspect of access control list for authorization. The extra credit portion of the homework does not need to be integrated with your last homework – rather they can be demonstrated using standalone producer/consumer applications.

References:
https://strimzi.io/
https://github.com/kubernetes-sigs/aws-ebs-csi-driver

Submission

The submission for this assignment should be through the blackboard website. I expect a zipped package containing the source files, configuration files, such as YAML files, and any additional packages, scripts, or files that you used. I also require a readme file which contains installation and setup instructions, including references, of the tools you used so that the TA and I can replicate your steps and deploy and run the assignment. I also expect AWS URL of your homepage as part of readme file.

Please add the link of your application to your website on Amazon S3, that you created in the Part1 section of this assignment. Also, provide the URL of your homepage as well as of the application deployed on Kubernetes in readme file as part of your HW submission on the class blackboard. Please provide a video recording demonstrating the working of every part of your application and make it a part of your submission. In addition, schedule a meeting with the GTA or the professor to demo your work.

NOTE: A late assignment carries a 10% late penalty for each week it is late. Assignments are NOT accepted after being 2 weeks late. Make sure your or your group's name is on every programming artifacts so we know who it belongs to. For every source file, please include comments at the top of the program describing what the program does. This only needs to be 1 or 2 sentences. Be sure to test access and functionality to your submission before the due date.

Grading:

The following areas will be used in the basic grading of these projects:

- Does system meet the functional and submission requirements: 85 Points
- Does the assignment run without errors: 15 Points

Instant Point Deductions:

I reserve the right to deduct points instantly for the following reasons:

- The source, or binary, files are not included in the package.
- The readme file is not included in the package.
- The program doesn't run due to errors in the code.
- I can't figure out how to use the assignment, and instructions are left out.