Apache Kafka for Event-Driven Spring Boot

Duration: 5 days (8hrs/day)

Prerequisite: --

- Basic System Administration
- Basics of Java

Course Objective:
Learn the fundamentals and basic concepts of OpenShift that you will need to build a production ready OpenShift cluster and get started with deploying and managing Application.

Lab Requirement: -- Koenig DC

Module 1: Introduction

- Introduction to Microservice
- Microservice vs Monolithic application
- Microservices Communication
- Event-Driven Architecture with Apache Kafka
- Apache Kafka for Microservices
- Messages and Events in Apache Kafka
- Kafka Topics & Partitions
- Ordering of Events in Apache Kafka

Module 2: Apache Kafka Broker

- What is Kafka Broker
- Leader and Follower roles. Leadership balance
- Download Apache Kafka
- Run Apache Kafka on Windows or in Docker
- Start single Apache Kafka broker with Kraft.
- Multiple Apache Kafka broker: Configuration Files
- Multiple Kafka Brokers: storage folders
- Starting Multiple Kafka Brokers with Kraft
- Stopping Kafka Brokers

Module 3: Kafka CLI Topics

- Introduction to Kafka Topic CLI
- Creating a New Kafka Topic
- List and Describe Kafka Topics
- Deleting a Topic

Module 4: Kafka CLI Producers

- Introduction to Kafka Producer CLI
- Producing a message without a key

- Send message as a Key-Value Pair

Module 5: Kafka CLI Consumers

- Introduction to Kafka Consumer CLI
- Consuming messages from Kafka topic from the beginning
- Consume new messages only
- Consuming Key:Value pair messages from Kafka Topic
- Consuming messages in Order

Module 6: Kafka Producer – Spring Boot Microservices

- Introduction to Kafka Producer
- Kafka Producer – Introduction to synchronous communication style
- Kafka Producer – Introduction to asynchronous communication style with use case
- Creating a New Spring Boot Project
- Kafka Producer configuration properties
- Creating Kafka Topic
- Run Microservices to create a new topic
- Creating Rest Controller
- Creating a Service Class
- Creating an Event Class
- Kafka Producer: Send message Asynchronously
- Kafka Asynchronous Send. Trying how it works
- Kafka Producer: Send message Synchronously
- Kafka Producer: Handle Exception in Rest Controller
- Kafka Producer: Logging Record Metadata Information
- Kafka Synchronous Send. Trying how it works

Module 6: Kafka Producer – Acknowledgements & Retries

- Introduction to Kafka Producer Acknowledgement and Retries
- Configure Producer Acknowledgements in Spring Boot Microservices
- The min.insync.replicas configuration
- Producer Retries
- Delivery & Request timeout
- Kafka Producer Spring Bean Configuration

Module 7: Apache Kafka Idempotent Producer

- Introduction to Idempotent Producer
- Enable Kafka Producer Idempotence in application.properties
- Enable Kafka Producer Idempotent in Spring Bean

KOENIG
step forward

Module 8: Kafka Consumer – Spring Boot Microservices

- Introduction to Kafka Consumer
- Creating a new spring Boot project
- Kafka Consumer Configuration properties
- KafkaEventListeners and KafkaHandler annotations
- Creating “core” module with required dependency
- Kafka Consumer Spring Bean Configuration
- Kafka Listener Container Factory

Module 9: Kafka Consumer – Handle Deserializer Errors

- Introduction to error Handling in Kafka Consumers
- Causing the Deserialization problem
- Kafka Consumer ErrorHandlingDeserilizer and its implementation

Module 10: Kafka Consumer – Dead Leter Topic(DLT)

- Introduction to DeadLeterTopic(DLT) in Kafka
- The DefaultErrorHandler and DeadLetterPublishingRecover classes
- Create and configure Kafka Template object

Module 11: Kafka Consumer – Exception and Retries

- Introduction to Exception Handling in Kafka consumer and Retries
- Creating retryable and not retryable exceptions
- Configure DefaultErrorHandler with a list of not retryable exceptions
- Trying how not retryable exception works
- Register RetryableException and define wait time interval
- Throwing a RetryableException
- Overview of a Destination Microservice
- Trying how retry works

Module 12: Kafka Consumer – Multiple consumers in a consumer group

- Introduction to kafka Consumer Group
- Rebalance and partition Assignment in Apache Kafka
- Assigning Microservices to a consumer group
- Trying how partitions assignment works
- Multiple consumers consuming messages from kafka topic

KOENIG
step forward

Module 13: Idempotent Consumer

- Introduction to Idempotent Consumer
- Unique id into message header
- Adding database -related dependencies
- Configure database connection details
- Creating JPA entity and Repository
- Storing unique message id in database table

Module 14: Apache Kafka Transactions

- Introduction to Transactions in Apache Kafka
- Enable Kafka Transactions in application properties and @Bean method
- Rollback transaction for specific exception
- Reading committed messages in Kafka Consumer
- Check how Kafka Transactions work
- Apache Kafka Local Transaction with Kafka Template

Module 15: Apache Kafka and Database Transactions

- Introduction to Kafka and Database Transactions
- Demo projects and Source projects
- Creating JPA Transactions manager
- Synchronized Transactions to save in Database
- Enable logging for both Kafka and JPA Transactions Managers
- Check Synchronized transactions work

Module 16: Integration Testing – Kafka Producer

- Introduction to System Under Test
- Test class annotations
- Creating an empty test method
- Different ways to execute test method
- Arrange, Act and Assert of Test method with implementation
- Consumer Configuration in a Test Class
- The SetUp() and TearDown() methods
- The Assert section
- Verify Producer Configuration Properties with Idempotent

Module 17: Integration Testing – Kafka Consumer

- Introduction to Method Under Test
- Creating a new test class with the ‘Arrange’ section
- Mocking section with ‘Act’ and ‘Assert’ sections
- Testing Kafka consumer

Module 18: Run Apache Kafka in a Docker Container

- Installation of Docker
- Initial Configuration of Kafka Docker Compose
- Configure Apache Kafka Kraft
- Apache Kafka Listeners Configuration
- Volumes as Kafka Docker Container
- Docker Compose and Environment file
- Starting single Kafka Server in Docker Container
- Execute Apache Kafka CLI Scripts in Docker Container
- Execute Apache Kafka CLI Commands in Docker from the Host Machine
- Execute Apache Kafka CLI Commands on the Host Machine
- Multiple Kafka Brokers in a Docker Compose File