Бауманский учебный центр «Специалист»

Программа курса Apache Kafka для Python - разработчиков и Data Science инженеров

Код курса: **КАФПИТ**

Тема	Ак. часов
Модуль 1. Введение в экосистму Apache Kafka • **Особенности событийно-ориентированной архитектуры.** Когда нужен Kafka, а когда RabbitMQ (сравнение). • **Ключевые концепции:** Сообщения, Топики, Партиции, Смещения (Offset), Репликация. • **Обзор Python-клиентов:** Сравнение kafka-python (стабильный, низкоуровневый) vs confluent-kafka. • **Подготовка окружения:** Установка и настройка Apache Kafka через Docker (dockercompose).	**4**
Модуль 2. Kafka Producer API в Python • **Структура ProducerRecord:** Ключ, значение, заголовки (headers) в Python. • **Асинхронная отправка:** Методы send() и flush(). Обработка колбэков (Callbacks). • **Пользовательское партицирование:** Реализация интерфейса Partitioner на Python. • **Конфигурация:** acks, retries, compression.type (gzip, snappy). • **Live Coding:** Написание продюсера, который читает данные из CSV и отправляет их в Kafka с пользовательской логикой распределения по партициям.	**2**
Модуль 3. Kafka Consumer API в Python • **Группы потребителей (Consumer Groups):** Стратегии распределения партиций (Range, RoundRobin, Cooperative). • **Rebalance Listeners:** Как правильно закрывать ресурсы и сохранять состояние при перебалансировке. • **Управление смещениями:** Ручное и автоматическое коммиты (enable_auto_commit, commit()). • **Чтение данных:** Цикл poll(), работа с ConsumerRecord. • **Live Coding:** Реализация консьюмера и сохранения прогресса в PostgreSQL.	**2**
Модуль 4. Сериализация, Десериализация и Pandas • **Форматы данных:** JSON, Avro, Protobuf. Сравнение эффективности в Python. • Использование сериализации и десериализации объектов при взаимодействии с Apache Kafka. • Интеграция Apache Kafka с модулем Pandas. • **Admin API:** Управление топиками, конфигурациями и партициями из Python-скриптов. • **Проект:** Разработка микросервиса, который принимает данные из API, обогащает их через Pandas и отправляет в Kafka.	**4**
Модуль 5. Гарантии доставки и Транзакции • **Idempotent Producer:** Настройка enable.idempotence=True в Python. • **Exactly Once Semantics (EOS):** Транзакционная запись. Инициализация транзакций (init_transactions), начало и фиксация. • **Consumer-транзакции:** Чтение committed смещений, обработка read_committed данных.	**2**

Модуль 6. Kafka Streams на Python: Faust • **Введение в Faust:** Создание потоковых приложений (StreamProcessor). • Работа с таблицами состояний (Tables). • **Хранение состояния:** Рокс-базы (RocksDB) в Python контексте.	**2**
Модуль 7. Schema Registry и Avro для Python • **Структура сообщений:** Почему плохо жить без схемы. • **Confluent Schema Registry:** Развертывание в Docker.	**1**
Модуль 8. Инструменты управления и мониторинга • **Confluent REST API:** Управление кластером из Python (requests). • **Метрики в Python:** Экспорт метрик продюсера и консьюмера. • **Мониторинг логов:** Как отслеживать состояние ребалансировки и ошибки сериализации.	**1**
Модуль 9. Расширенные возможности и Интеграции • **Kafka Connect:** Запуск коннекторов. Использование REST API для управления коннекторами. • **ksqlDB:** Выполнение SQL-запросов к потокам данных из Python. • **Работа с большими данными:** Чтение из Kafka в Pandas для пакетной аналитики.	**2**
Модуль 10. Итоговая работа: "Аналитический пайплайн"	**4**
Аудиторная нагрузка в классе с преподавателем	**24 ак.ч. + 24 ак.ч. самостоятельно**

По окончании обучения на курсе проводится итоговая аттестация. Аттестация проводится в виде теста на последнем занятии или на основании оценок практических работ, выполняемых во время обучения на курсе.

Контакты центра

Для частных лиц
+7 (495) 232-32-16

Для организаций
+7 (495) 780-48-44

E-mail
info@specialist.ru