# Travail Pratique: Introduction à Apache Kafka avec Python

## Objectifs

Ce laboratoire a pour objectif de :  
- Comprendre les bases de Apache Kafka.  
- Installer et configurer un environnement Kafka.  
- Réaliser un exercice pratique utilisant Kafka avec Python pour simuler une application de messagerie.

## Prérequis

Avant de commencer ce laboratoire, assurez-vous de :  
- Avoir Python 3.x installé sur votre machine.  
- Avoir Docker installé (option recommandé pour exécuter Kafka facilement).  
- Connaître les bases de la programmation Python.

## 1. Introduction à Kafka

Apache Kafka est une plateforme distribuée de streaming utilisée pour gérer les flux de données en temps réel.   
Il permet de produire, consommer, stocker et traiter des messages efficacement dans des environnements distribués.  
  
Concepts clés :  
- **Topic** : Conteneur logique pour les messages.  
- **Producer** : Composant qui envoie des messages à un topic.  
- **Consumer** : Composant qui lit les messages depuis un topic.  
- **Broker** : Serveur Kafka responsable du stockage des messages.

## 2. Préparation de l'environnement

Ce laboratoire utilise Docker pour configurer Kafka et ZooKeeper.

### Étapes pour configurer l'environnement

1. Créez un fichier nommé *docker-compose.yml* avec le contenu suivant :

version: "3.8"

services:

  zookeeper:

    image: confluentinc/cp-zookeeper:latest7

    environment:

      ZOOKEEPER\_CLIENT\_PORT: 2181

    ports:

      - "2181:2181"

 kafka:

    image: confluentinc/cp-kafka:latest

    depends\_on:

      - zookeeper

    environment:

      KAFKA\_BROKER\_ID: 1

      KAFKA\_ZOOKEEPER\_CONNECT: zookeeper:2181

      KAFKA\_ADVERTISED\_LISTENERS: PLAINTEXT://localhost:9092

      KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP: PLAINTEXT:PLAINTEXT

      KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR: 1

    ports:

      - "9092:9092"

2. Lancez les services avec la commande : `docker-compose up -d`

3. Assurez-vous que Kafka est en cours d'exécution en vérifiant les logs : `docker-compose logs -f kafka`

## 3. Exercice : Application de messagerie avec Kafka et Python

Dans cet exercice, nous allons utiliser Kafka pour simuler une application de messagerie simple.  
Vous écrirez un **Producer** pour envoyer des messages à un topic et un **Consumer** pour lire ces messages.

### Étape 1 : Installer les dépendances

1. Installez la bibliothèque Kafka pour Python :

pip install kafka-python

### Étape 2 : Créer un Producer

Le producer enverra des messages à un topic nommé `messages`. Créez un fichier `producer.py` avec le contenu suivant :

from kafka import KafkaProducer

from time import sleep

producer = KafkaProducer(bootstrap\_servers='localhost:9092')

while True:

    message = input("Entrez un message à envoyer : ")

    producer.send('messages', message.encode('utf-8'))

    print(f"Message envoyé : {message}")

    sleep(1)

### Étape 3 : Créer un Consumer

Le consumer lira les messages du topic `messages`. Créez un fichier `consumer.py` avec le contenu suivant :

from kafka import KafkaConsumer

# Initialisation du consumer pour écouter le topic 'messages'

consumer = KafkaConsumer(

    'messages',  # Nom du topic

    bootstrap\_servers='localhost:9092',  # Serveur Kafka

    auto\_offset\_reset='earliest',  # Lire depuis le début si aucun offset n'est présent

    enable\_auto\_commit=True,  # Confirmer automatiquement les messages lus

    group\_id='group1'  # Nom du groupe de consommateurs

)

print("En attente de nouveaux messages...")

# Boucle infinie pour écouter les messages

try:

    for message in consumer:

        print(f"Message reçu : {message.value.decode('utf-8')}")

except KeyboardInterrupt:

    print("\nArrêt du consumer.")

finally:

    consumer.close()

### Étape 4 : Tester votre application

1. Lancez le consumer en exécutant `python consumer.py`.  
2. Dans un autre terminal, lancez le producer avec `python producer.py`.  
3. Envoyez plusieurs messages depuis le producer et observez-les s'afficher dans le consumer.

## 4. Conclusion

Dans ce laboratoire, vous avez appris à configurer Kafka et à interagir avec lui en utilisant Python.  
Vous avez également exploré les concepts fondamentaux de Kafka à travers une simulation simple d'une application de messagerie.