**Filière : IITE3**

**Module : Architecture et Infrastructure Big Data**

**Année universitaire : 2023/2024**

**Travaux Pratique 8 :**

**Apache Kafka+ Spark Streaming + Flume**



**Rédigé par : Encadré par :**

**HMAYDA Abdessamad Pr. Kalloubi Fouad**

Table des matières

[**I.** **Apache Kafka :** 3](#_Toc153054529)

[**1.** **Mise en place** 3](#_Toc153054530)

[**2.** **Introduction :** 4](#_Toc153054531)

[**a.** **Création d’un topic « ensademo » :** 4](#_Toc153054532)

[**b.** **S’assurer que le topic a été créé :** 4](#_Toc153054533)

[**c.** **Démarrage du producteur :** 4](#_Toc153054534)

[**d.** **Démarrage du consommateur :** 4](#_Toc153054535)

[**e.** **Affichage du contenu de notre topic :** 4](#_Toc153054536)

[**f.** **Travail à faire :** 4](#_Toc153054537)

[**II.** **Spark streaming :** 5](#_Toc153054538)

[**1) Lancement de SparkStreamingContext** 5](#_Toc153054539)

[1)- Tout d’abord allez au terminal de votre namenode et tapez : **hdfs dfs -chmod -R 777 /** 5](#_Toc153054540)

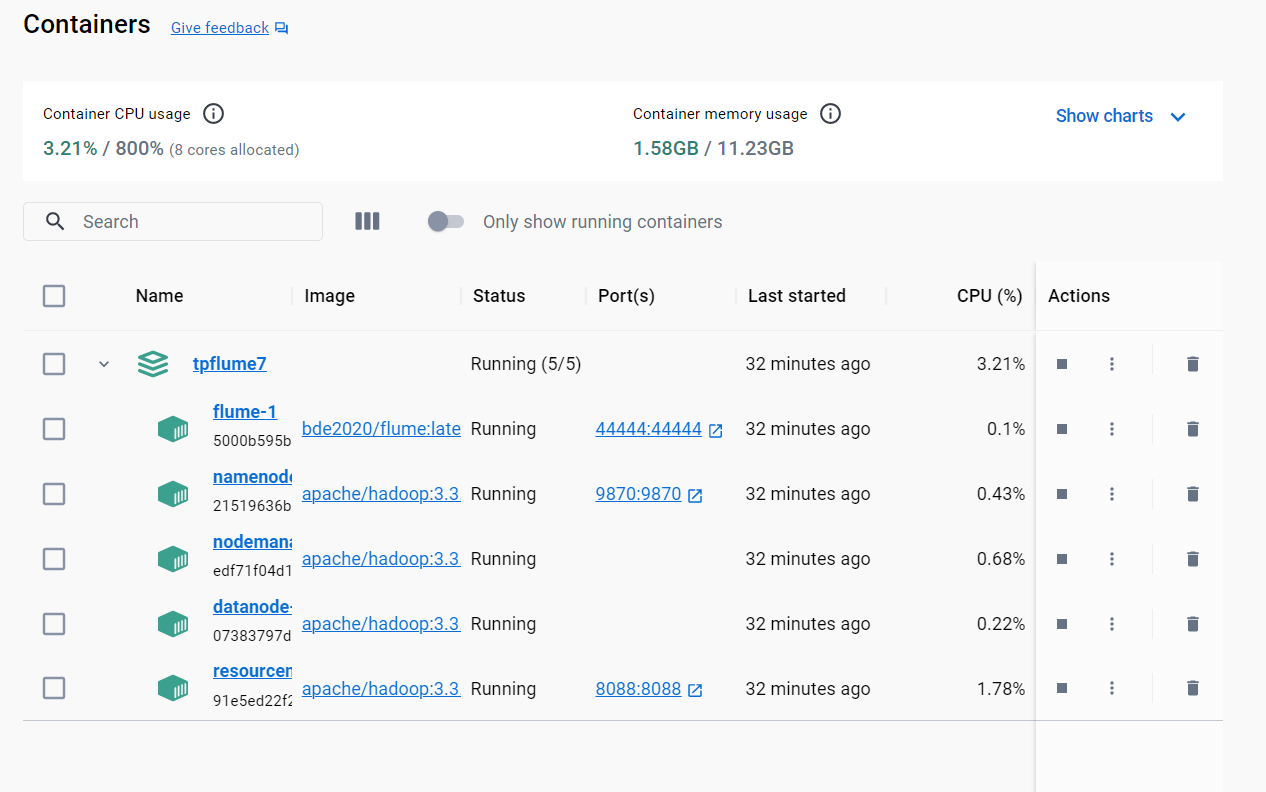
[**2)-Développement d’une application « wordCount » en utilisant Spark Streaming :** 6](#_Toc153054541)

# **Apache Kafka :**

## **Mise en place**

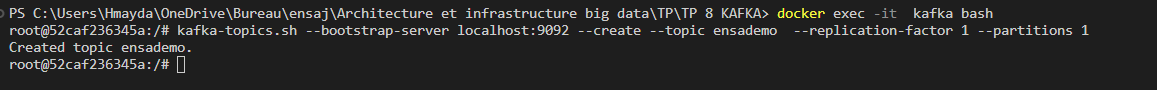
-Cloner ce projet, et exécuter tapez « docker compose up » à l’intérieure du dossier du projet

 - tapez « docker ps » pour s’assurer que les conteneurs marchent bien comme l’image d’apres docker desktop montre



## **Introduction :**

### **Création d’un topic « ensademo » :**



### **S’assurer que le topic a été créé :**



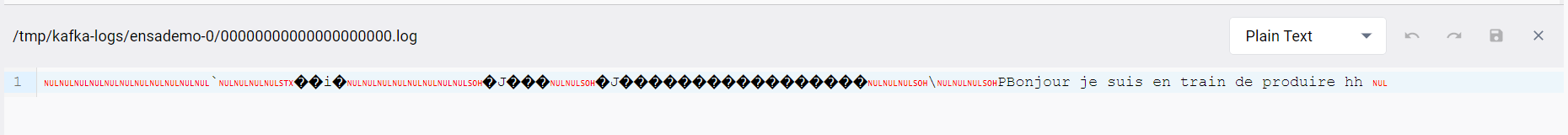
### **Démarrage du producteur :**



### **Démarrage du consommateur :**



### **Affichage du contenu de notre topic :**



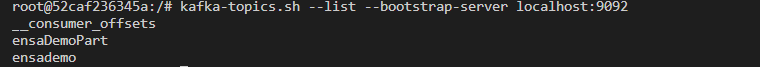
### **Travail à faire :**

#### **Créer un topic « ensaDemoPart » :**

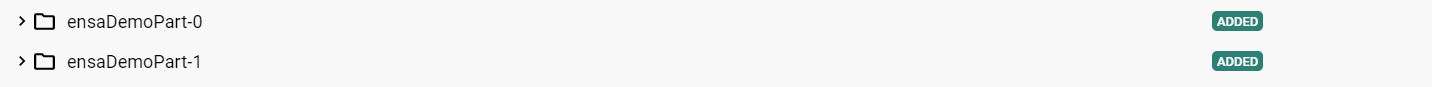
-Cette fois on va créer le topic avec un nombre de partitions égal à 2 :



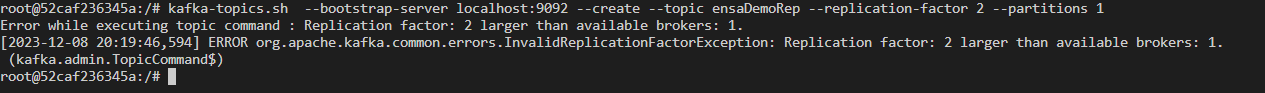
-la liste des topics



* Puisque le nombre de partitions de ensaDemoPart est 2, donc au niveau de « /tmp/kafka-logs » on aura deux partitions des logs de ce topic, comme indiqué dans l’image suivante :



#### **Créer un topic avec une facteur de réplication qui vaut 2 :**



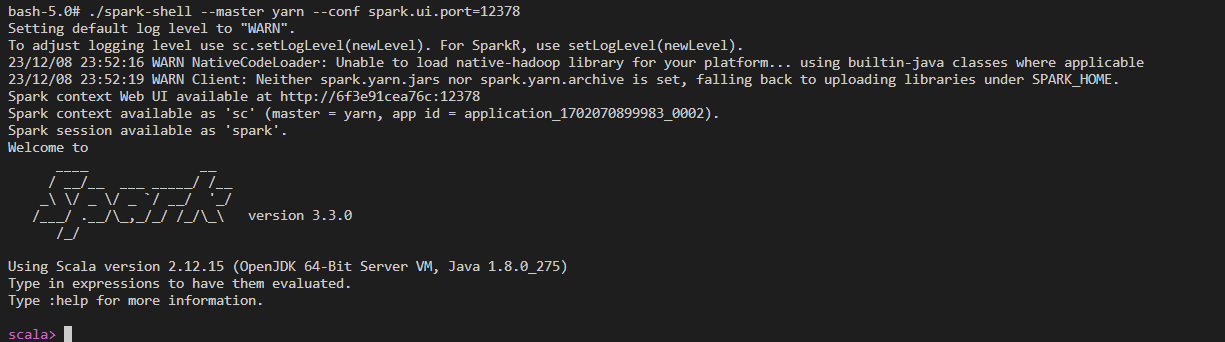
Le message d'erreur Facteur de réplication : 2 plus grand que les courtiers(brokers) disponibles : 1 indique qu’on essaye de créer un sujet Kafka avec un facteur de réplication supérieur au nombre de courtiers disponibles dans notre cluster Kafka.

# **Spark streaming :**

## **1) Lancement de SparkStreamingContext**

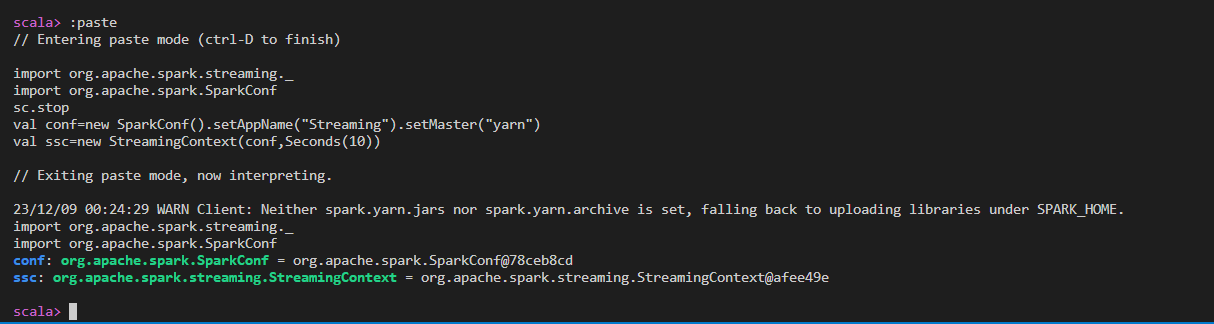
# 1)- Tout d’abord allez au terminal de votre namenode et tapez : **hdfs dfs -chmod -R 777 /**

2)- lancez le mode interactif de spark :



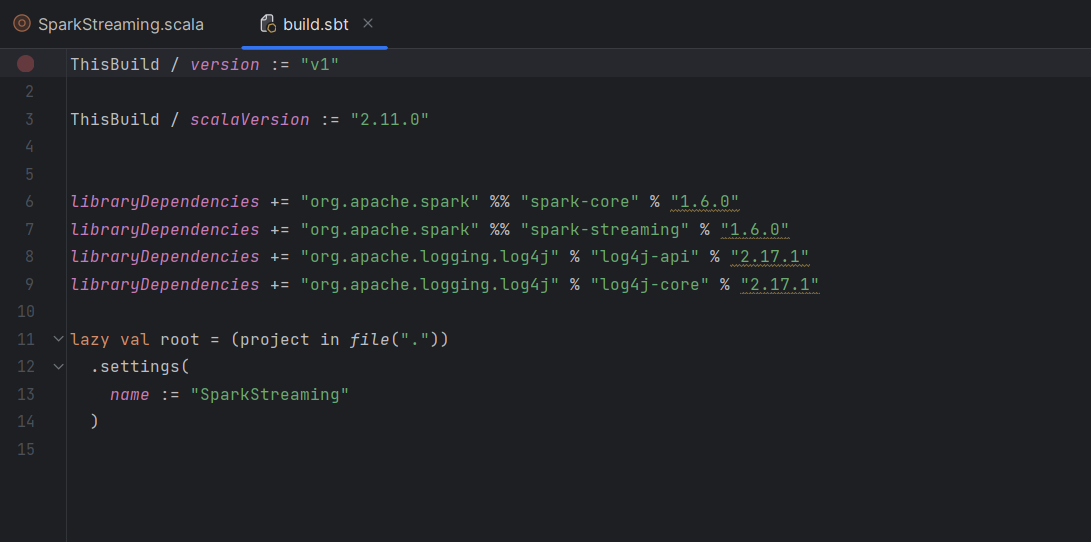
* Remarques :
  + Spark streaming est utilisé pour traiter les donnees en mode streaming, il s’appuie sur un service web appelé StreamingContext (on utilisera netcat dans notre cas pour simuler un service web).
  + StreamingContext exécute un traitement perpétuel de données à intervalles réguliers
  + Nous ne pouvons pas avoir plusieurs contextes en meme temps, donc on doit arrêter sparkContexte avant de lancer sparkStreaming

3) -arrêtez SparkContext et lancez SparkStreamingContext :

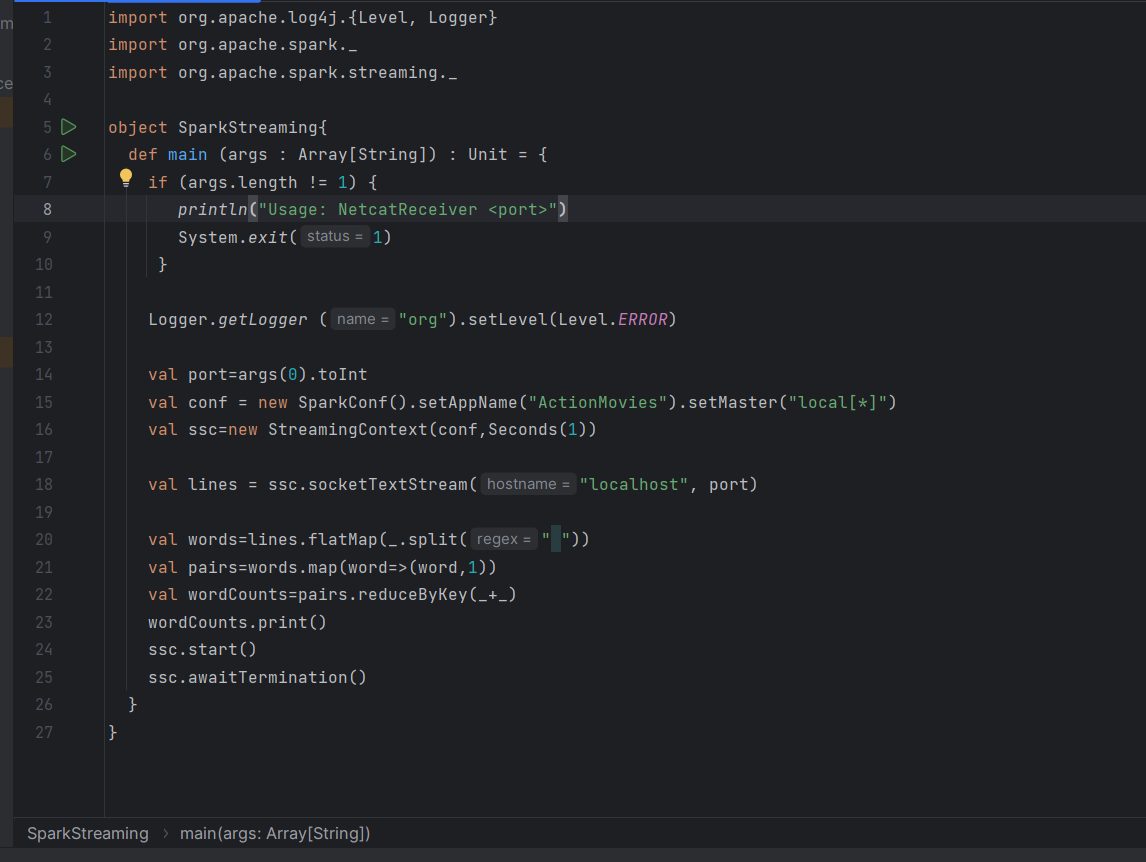


## **2)-Développement d’une application « wordCount » en utilisant Spark Streaming :**

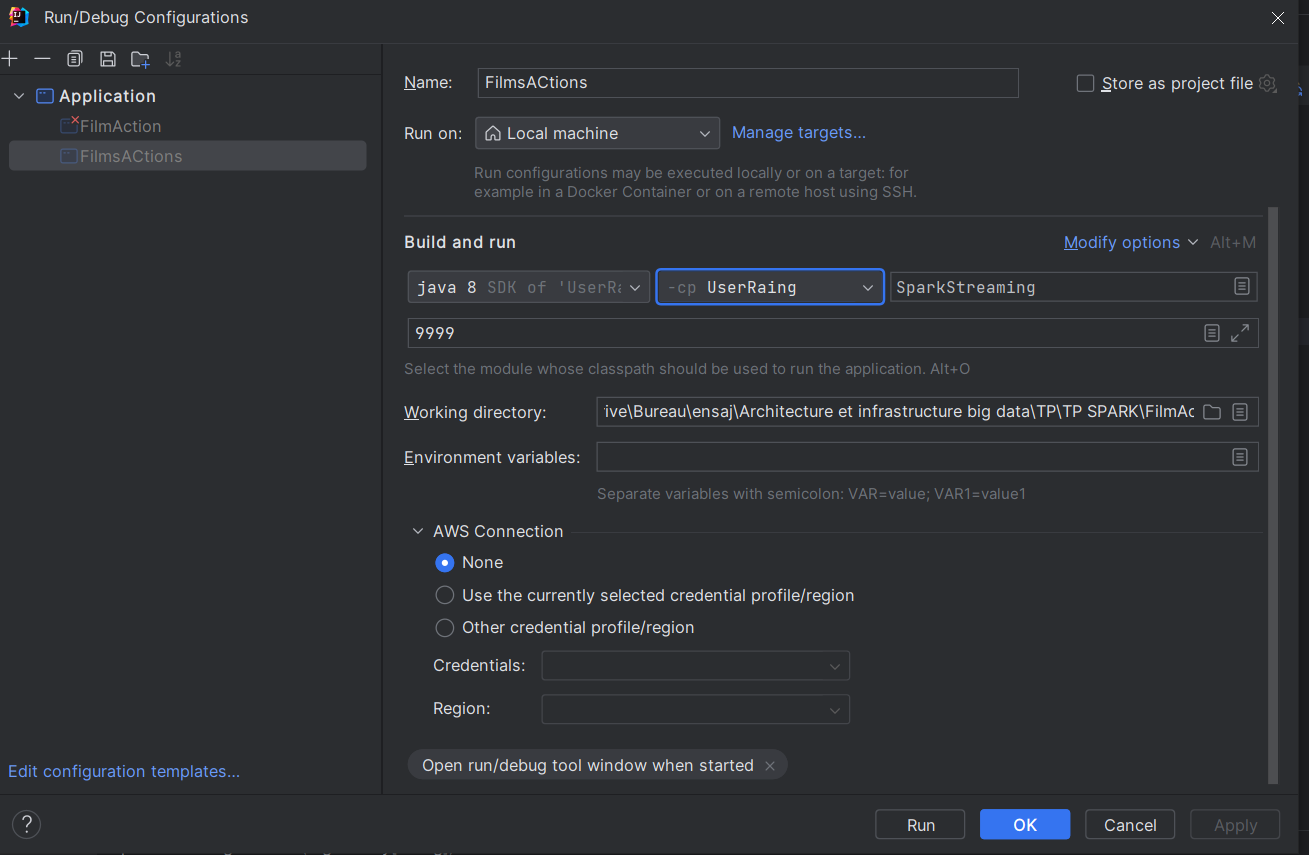
Tout d’abord nous allons créer une application Spark en utilisant Scala(2.11.0) et sbt (1.3.3) et apres importer spark et spark-streaming



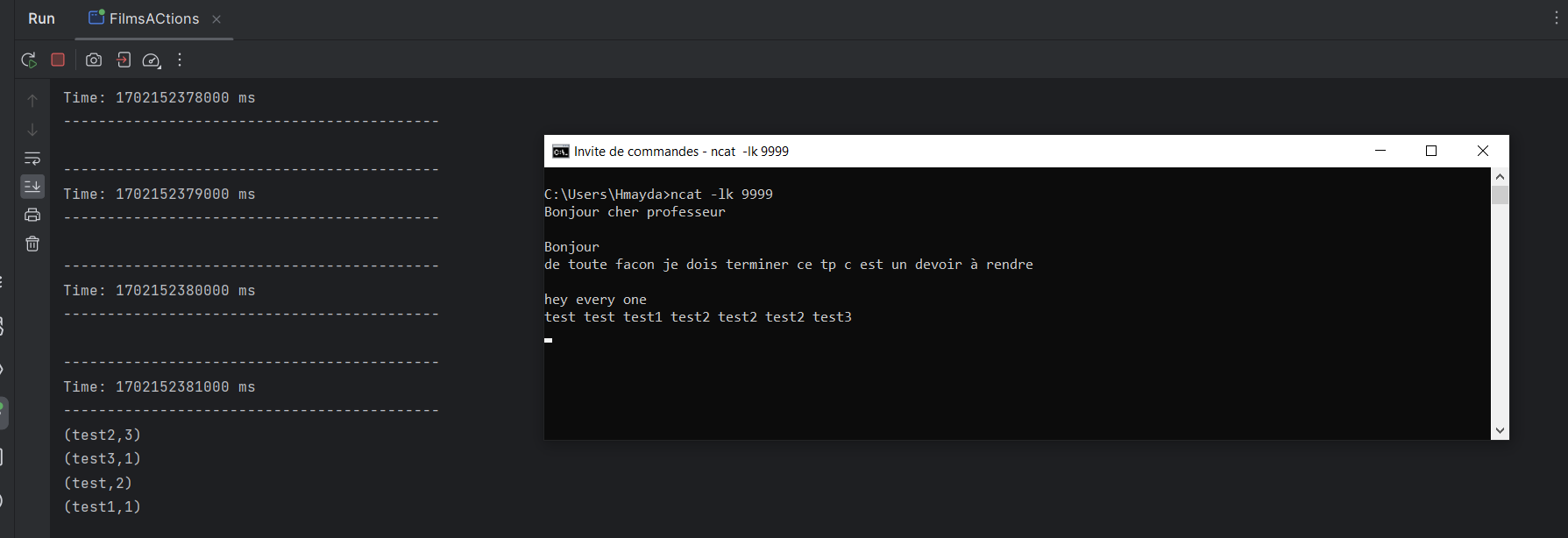
Voici le code de notre application



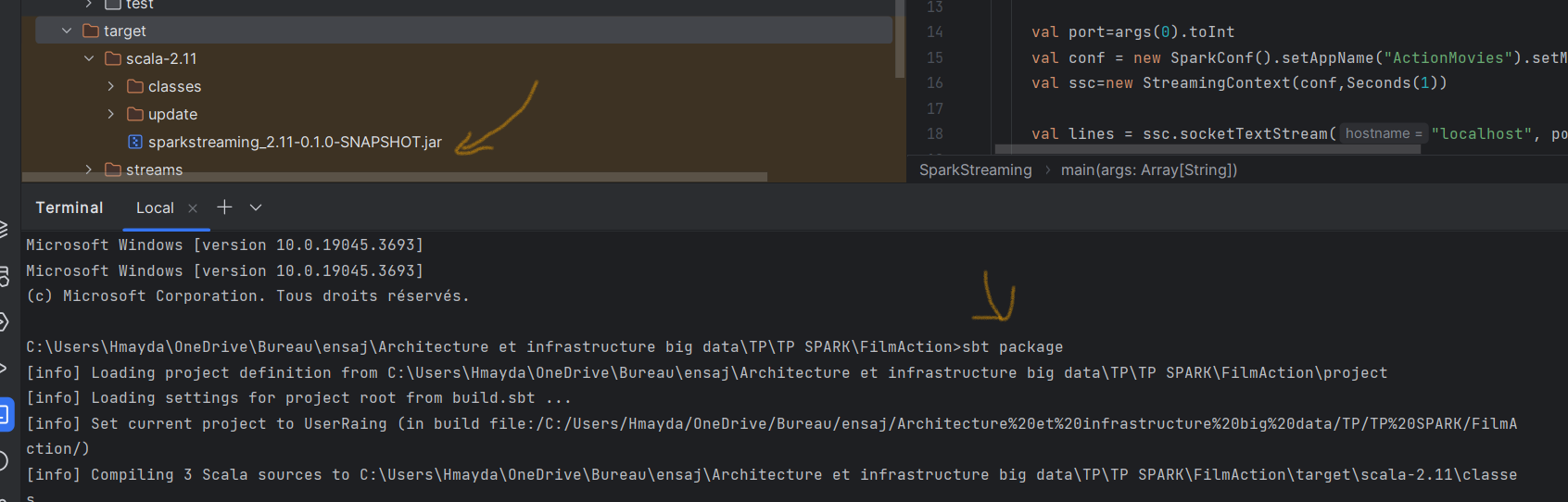
Exécuter cette application en passant 9999 comme argument



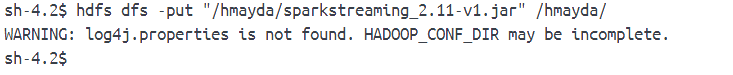
Également nous allons simuler un service web en utilisant netcat :



-donc notre application repond à nos besoin , on doit maintenant contsruire le jar de cette application :



* Ensuite, on doit mettre le jar dans HDFS, pour le faire deplacez le vers le dossier du volume hmayda et exécuter cette commande de hdfs :

- Lancez le jar avec spark-submit



-utiliser netcat et envoyez des messages sur le port 9999 et voici le résultat du traitement effectué par l’application spark

