



**UNIVERSITÉ IBN ZOHR**  
**FACULTÉ DES SCIENCES**  
**DÉPARTEMENT D'INFORMATIQUE**

**Master Spécialisé**  
Systèmes Informatiques Distribués & Big Data

---

**Guide de Configuration**  
**Projet d'Analyse En Temps Réel des données de Trafic Aérien**

---

**Réaliser par :**

Boujarfaoui Ayman

En-nahel Aissam

Ayoub Ellaouzi

El-adarissi Abdelaziz

**Encadré par :**

Professeur Mr.Karim Afdel

Année universitaire 2022/2023



# Table des matières

<b>1</b>	<b>Configuration d'événement</b>	<b>2</b>
1.1	Téléchargement . . . . .	2
1.2	Installation . . . . .	6
1.3	Modifications dans les variables d'environnement . . . . .	11
1.4	Kibana et Elasticsearch . . . . .	16
<b>2</b>	<b>Test d'Application</b>	<b>18</b>

# Table des figures

# Introduction

Notre projet, intitulé "Analyse en temps réel des données de trafic aérien à l'aide d'Apache Spark et Kafka Framework", vise à mettre en œuvre un système de traitement en temps réel des données de trafic aérien en utilisant deux technologies : Apache Kafka et Apache Spark.

L'objectif principal est de collecter les données de trafic aérien en temps réel à partir de l'API "Flight Tracker API and Data - Aviation database and API" et de les analyser en continu pour fournir des informations en temps réel sur le trafic aérien.

Apache Kafka joue un rôle crucial en tant que système de messagerie distribuée, facilitant ainsi la collecte et la gestion des flux de données en temps réel. Les données de trafic aérien collectées seront acheminées vers les topics correspondants, où elles seront prêtes à être traitées par Apache Spark. D'autre part, Apache Spark est un framework de traitement de données en temps réel offrant des fonctionnalités avancées de traitement distribué et d'analyse. En configurant Apache Spark Streaming, nous pourrions mettre en place des tâches de traitement en temps réel pour analyser les données de trafic aérien provenant de Kafka.

La relation étroite entre Apache Kafka et Apache Spark est cruciale dans ce projet, car Kafka agit en tant que fournisseur de données en temps réel pour Spark. En utilisant Kafka comme source de données pour Spark Streaming, nous assurons un flux continu de données pour une analyse en temps réel efficace.

Ce projet offre une opportunité passionnante de mettre en pratique nos connaissances sur Apache Spark et Kafka Framework, en développant un système de traitement en temps réel fonctionnel pour l'analyse des données de trafic aérien.

# Chapitre 1

## Configuration d'événement

### 1.1 Téléchargement

- Télécharger JDK 1.8 dans le site officiel d'oracle  
<https://www.oracle.com/java/technologies/downloads/>

Java 8   Java 8 Enterprise Performance Pack   **Java 11**

---

**Java SE Development Kit 11.0.20**

Java downloads   Tools and resources   Java archive

Commercial license and support are available for a low cost with [Java SE Universal Subscription](#).

JDK 11 software is licensed under the [Oracle Technology Network License Agreement for Oracle Java SE](#).

JDK 11.0.20 [checksums](#)

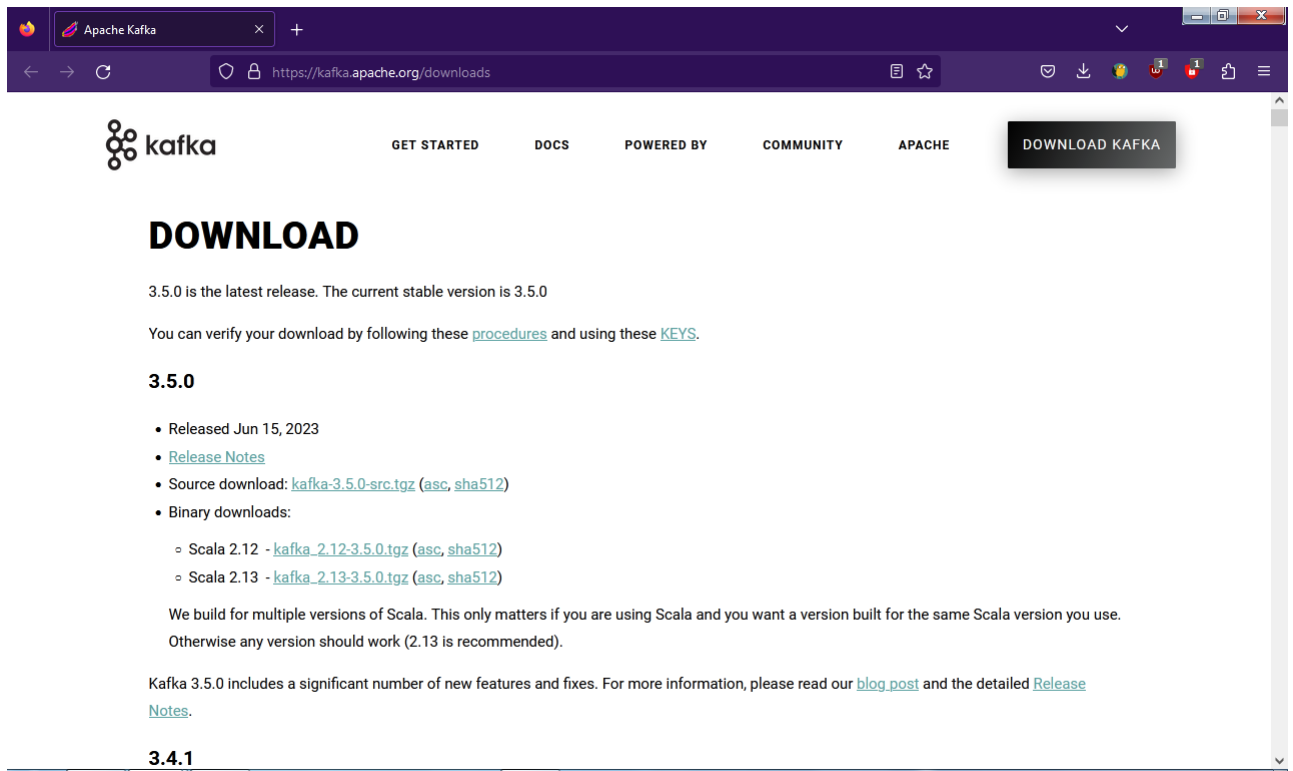
Linux   macOS   Solaris   **Windows**

Product/file description	File size	Download
x64 Installer	141.39 MB	<a href="#">jdk-11.0.20_windows-x64_bin.exe</a>
x64 Compressed Archive	159.14 MB	<a href="#">jdk-11.0.20_windows-x64_bin.zip</a>

[Documentation Download](#)

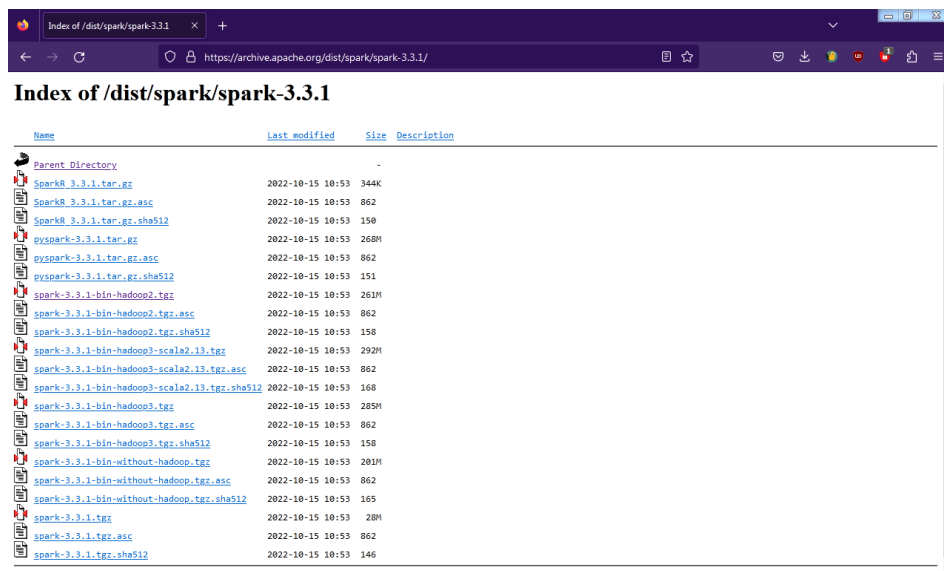
- Téléchargement d'Apache Kafka version 3.5.0 (kafka\_2.12-3.5.0)  
<https://kafka.apache.org/downloads>

## Chapitre 1 Configuration d'événement



- Téléchargement d'Apache Spark (spark-3.3.1)

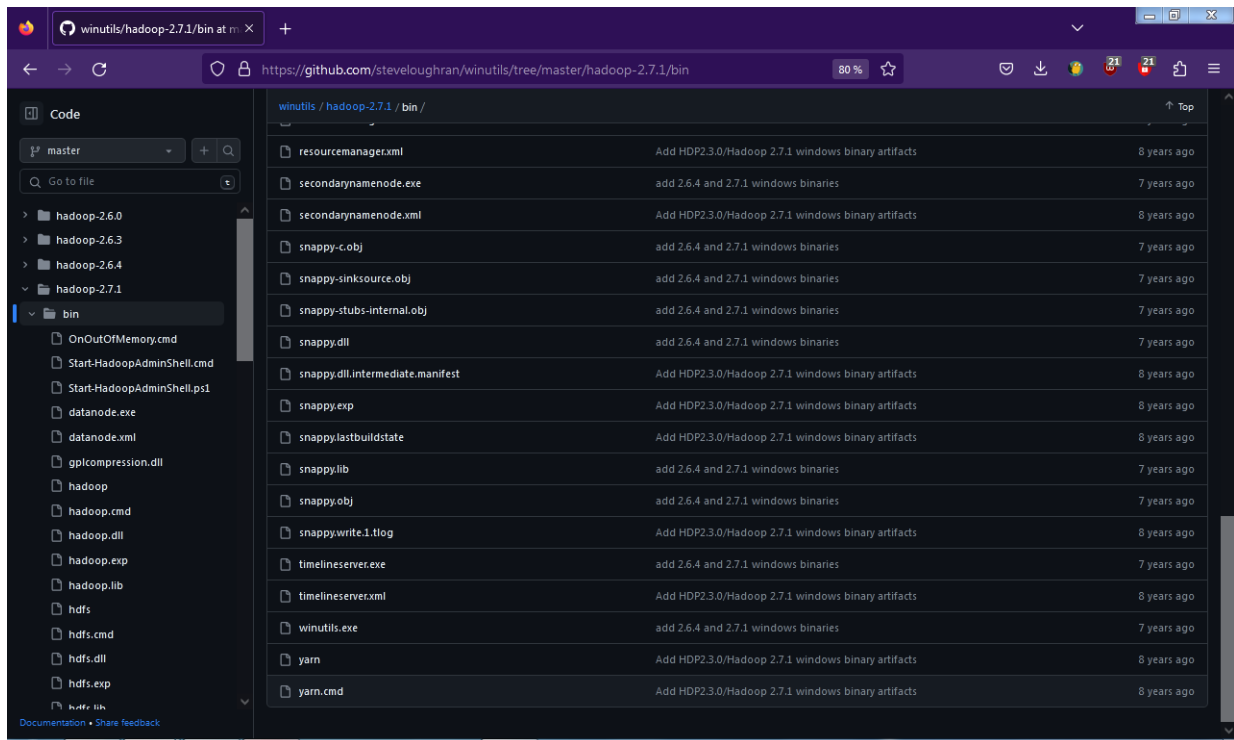
<https://archive.apache.org/dist/spark/spark-3.3.1/>



- Téléchargement de la version appropriée de winutils.exe pour Hadoop

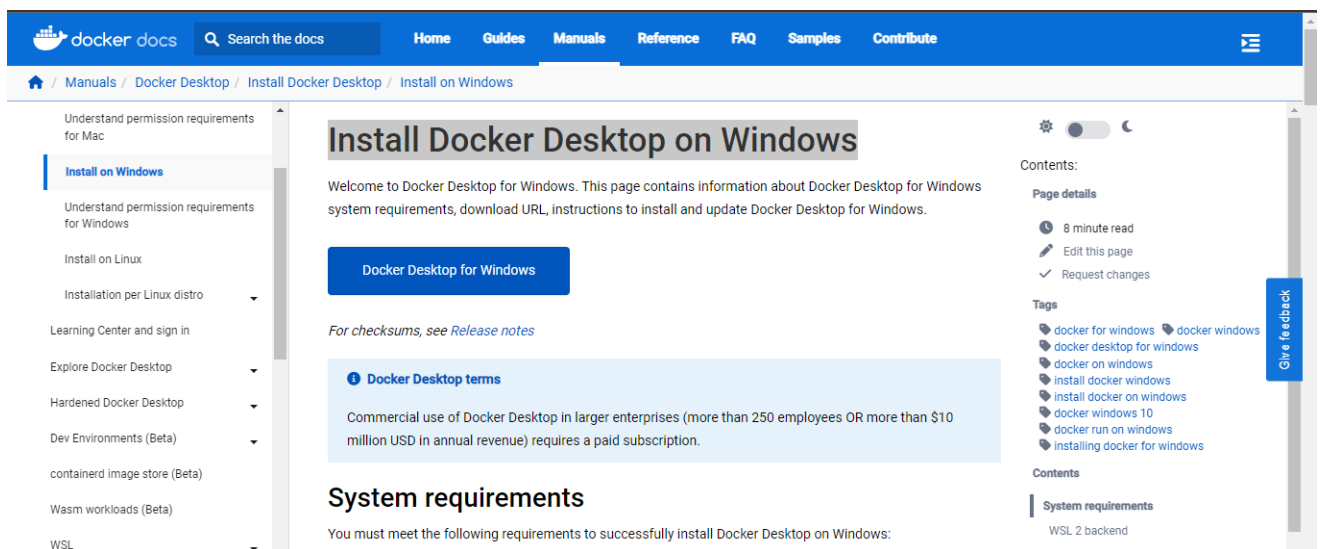
<https://github.com/stevcloughran/winutils/blob/master/hadoop-2.7.1/bin/winutils.exe>

## Chapitre 1 Configuration d'événement



- Install Docker Desktop on Windows

<https://docs.docker.com/desktop/install/windows-install/>



- Télécharger le package de mise à jour du noyau Linux WSL

<https://learn.microsoft.com/fr-fr/windows/wsl/install-manual#step-4---/download-the-linux-kernel-update-package>



## Chapitre 1 Configuration d'événement

- Documentation WSL
- > Vue d'ensemble
- > Installer
  - Installer WSL 2
  - Étapes d'installation manuelle pour les anciennes versions**
  - Installation sur Windows Server
- > Tutoriels
- > Concepts
- > Procédures
- Forum Aux Questions (FAQ)
- Résolution des problèmes
- > Notes de publication
- Télécharger le PDF

```
dim.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

## Redémarrez votre ordinateur pour terminer l'installation de WSL et mettre à jour vers WSL 2.

# Étape 4 : télécharger le package de mise à jour du noyau Linux

- Téléchargez le dernier package :
  - [Package de mise à jour du noyau Linux WSL2 pour machines x64](#)

**Notes**

Si vous utilisez une machine ARM64, téléchargez le [package ARM64](#) à la place. Si vous n'êtes pas sûr du type de machine que vous avez, ouvrez l'invite de commandes ou PowerShell et entrez : `systeminfo | find "System Type"`.

**Avertissement :** Sur les versions non anglaises de Windows, vous devrez probablement modifier le texte recherché et traduire la chaîne « System Type ». Vous devrez peut-être aussi échapper les guillemets pour la commande find. Par exemple, en allemand `systeminfo | find '"Systemtyp"'`.

### Ressources supplémentaires

- [Documentation](#)
- [Configurer un environnement de développement WSL](#)
- Configurez un environnement de développement WSL à l'aide des meilleures pratiques de ce guide d'ensemble par étape...
- [Résolution des problèmes liés au sous-système Windows pour Linux](#)
- Cet article fournit des informations détaillées sur les erreurs et problèmes courants auxquels peuvent être confrontés les utilisateurs...
- [Comparaison des versions WSL](#)
- WSL 2 offre les avantages de WSL 1, mais utilise un noyau Linux réel, au lieu d'une couche de traduction comme WSL 1, ce qui...
- [Afficher 5 de plus](#)

- Install pyspark from pip  
`pip install pyspark`

```
C:\Windows\system32\CMD.exe - pip install pyspark
Microsoft Windows [version 10.0.19045.3324]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\Paname>pip install pyspark
Collecting pyspark
  Downloading pyspark-3.4.1.tar.gz (310.8 MB)
    ----- 310.8/310.8 MB 2.7 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting py4j==0.10.9.7
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
    ----- 200.5/200.5 KB 12.7 MB/s eta 0:00:00
Using legacy 'setup.py install' for pyspark, since package 'wheel' is not installed.
Installing collected packages: py4j, pyspark
  Running setup.py install for pyspark ...
```

## Test the PySpark Installation

```
C:\Windows\system32\CMD.exe - pyspark
Microsoft Windows [version 10.0.19045.3324]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\Paname>pyspark
Python 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022, 16:36:42) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/08/21 14:35:52 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Welcome to

  ____      _
 / ___|  __| | | |
 \___ \  | | | | | |
  ___) | | | | | | |
 |_____|_|_|_|_|_|_|

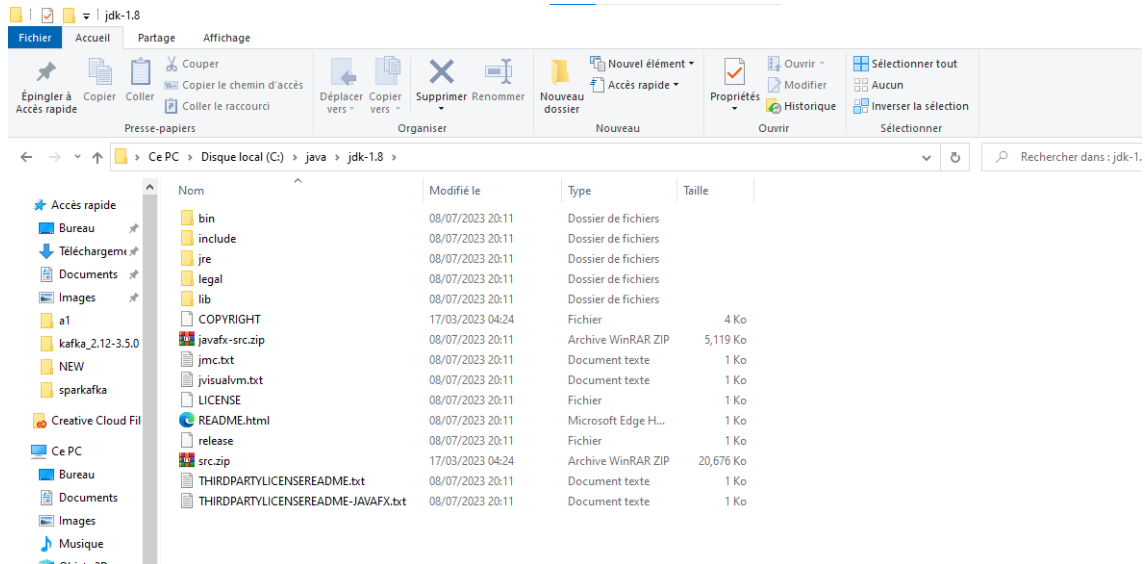
version 3.3.1

Using Python version 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022 16:36:42)
Spark context Web UI available at http://DESKTOP-UIF5V43:4040
Spark context available as 'sc' (master = local[*], app id = local-1692624955074).
SparkSession available as 'spark'.

>>> sc.version
'3.3.1'
>>>
```

## 1.2 Installation

- Après l'installation du Java **jdk-8u371-windows-x64.exe** dans **C :/Java/**



- Test JDK Version

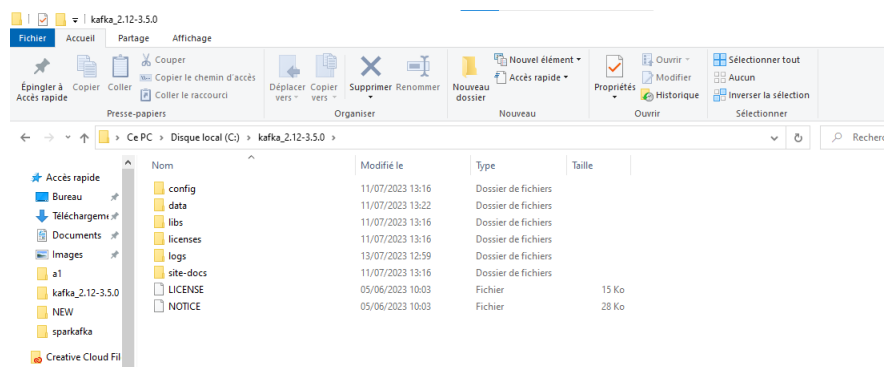
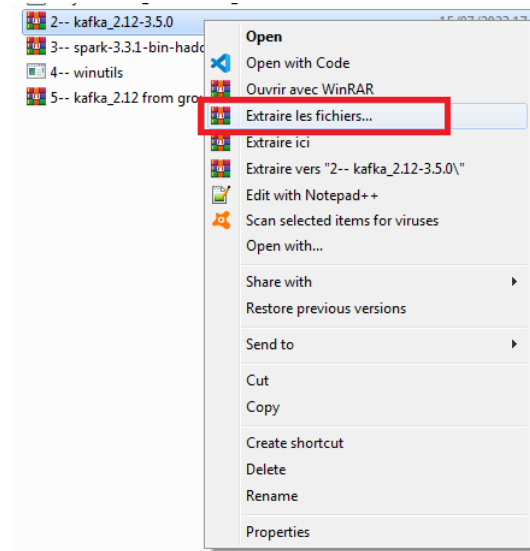
```
C:\Windows\system32\CMD.exe

C:\Users\Paname>java -version
java version "1.8.0_371"
Java(TM) SE Runtime Environment (build 1.8.0_371-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.371-b11, mixed mode)

C:\Users\Paname>
```

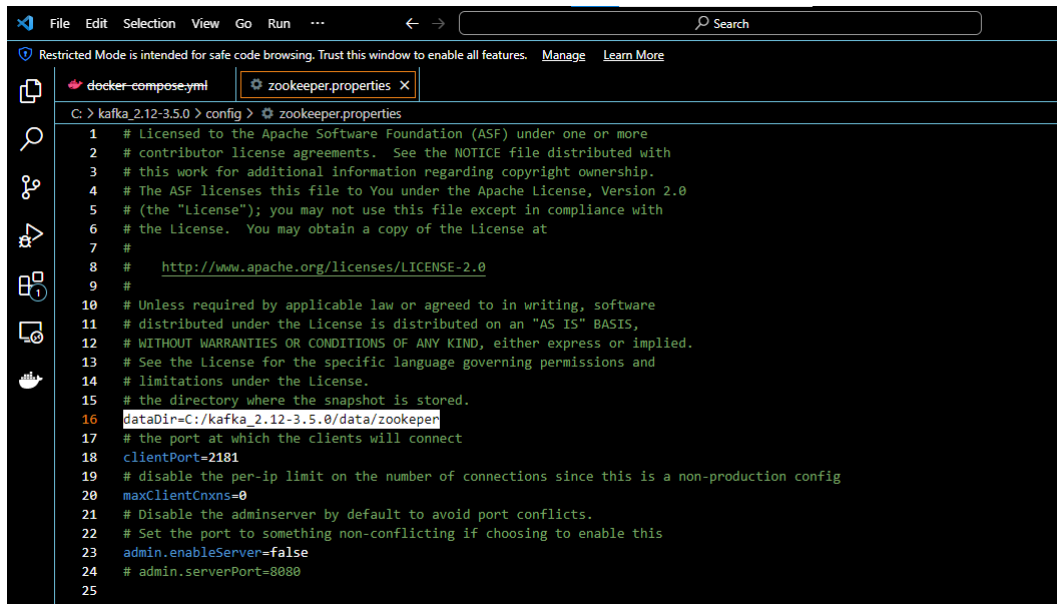
- Après On va décompresser l'archive kafka\_2.12-3.5.0.tgz dans le répertoire **C :/**

## Chapitre 1 Configuration d'événement



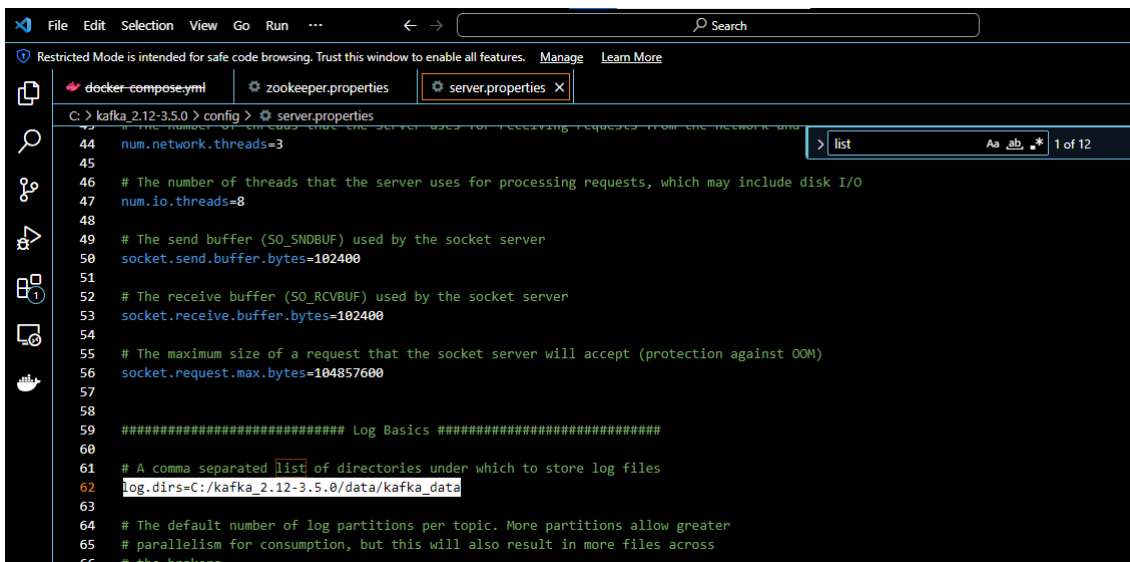
## Chapitre 1 Configuration d'événement

- On va modifier une ligne dataDir dans `C:\kafka\_2.12-3.5.0\config\zookeeper.properties` par `dataDir=C:/kafka_2.12-3.5.0/data/zookeeper`



```
1 # Licensed to the Apache Software Foundation (ASF) under one or more
2 # contributor license agreements. See the NOTICE file distributed with
3 # this work for additional information regarding copyright ownership.
4 # The ASF licenses this file to You under the Apache License, Version 2.0
5 # (the "License"); you may not use this file except in compliance with
6 # the license. You may obtain a copy of the License at
7 #
8 # http://www.apache.org/licenses/LICENSE-2.0
9 #
10 # Unless required by applicable law or agreed to in writing, software
11 # distributed under the license is distributed on an "AS IS" BASIS,
12 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 # See the license for the specific language governing permissions and
14 # limitations under the license.
15 # the directory where the snapshot is stored.
16 dataDir=C:/kafka_2.12-3.5.0/data/zookeeper
17 # the port at which the clients will connect
18 clientPort=2181
19 # disable the per-ip limit on the number of connections since this is a non-production config
20 maxClientCnxns=0
21 # Disable the adminserver by default to avoid port conflicts.
22 # Set the port to something non-conflicting if choosing to enable this
23 admin.enableServer=false
24 # admin.serverPort=8080
25
```

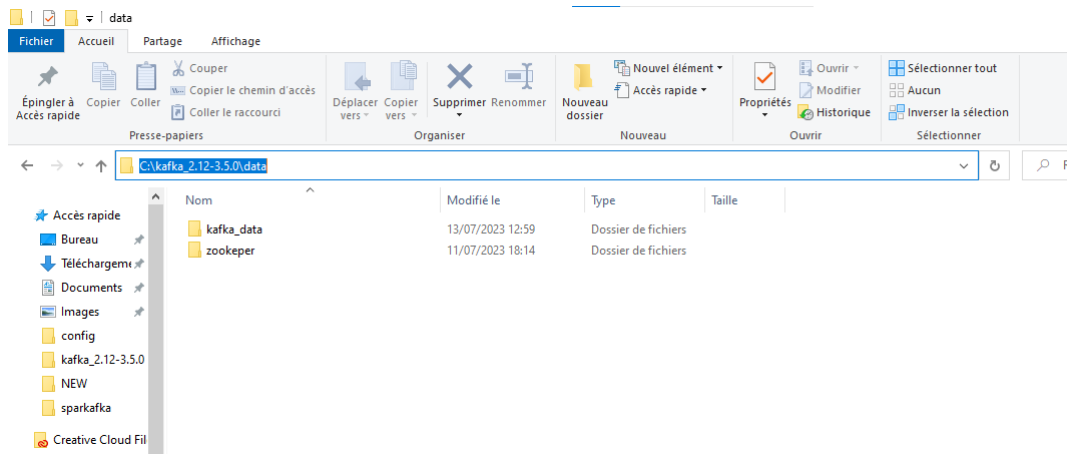
- Et On va modifier une ligne dans `C:/kafka/_2.12-3.5.0/config/server.properties` `log.dirs=C:/kafka_2.12-3.5.0/data/kafka_data`



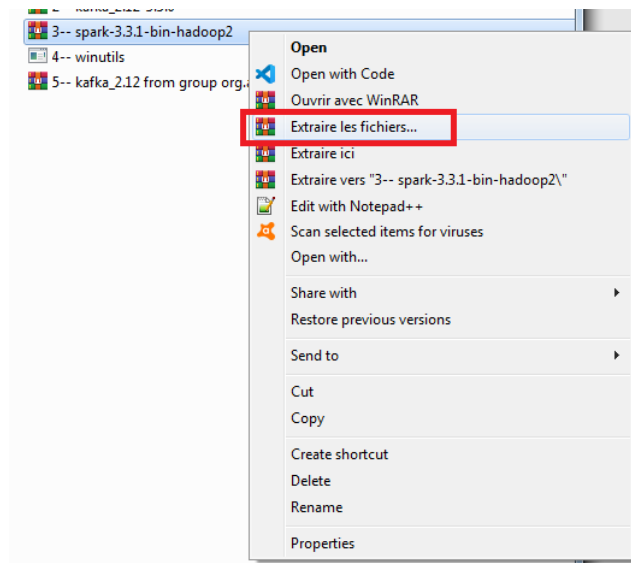
```
44 num.network.threads=3
45
46 # The number of threads that the server uses for processing requests, which may include disk I/O
47 num.io.threads=8
48
49 # The send buffer (SO_SNDBUF) used by the socket server
50 socket.send.buffer.bytes=102400
51
52 # The receive buffer (SO_RCVBUF) used by the socket server
53 socket.receive.buffer.bytes=102400
54
55 # The maximum size of a request that the socket server will accept (protection against OOM)
56 socket.request.max.bytes=104857600
57
58 ##### Log Basics #####
59
60 # A comma separated list of directories under which to store log files
61 log.dirs=C:/kafka_2.12-3.5.0/data/kafka_data
62
63 # The default number of log partitions per topic. More partitions allow greater
64 # parallelism for consumption, but this will also result in more files across
65 # the brokers.
66
```

## Chapitre 1 Configuration d'événement

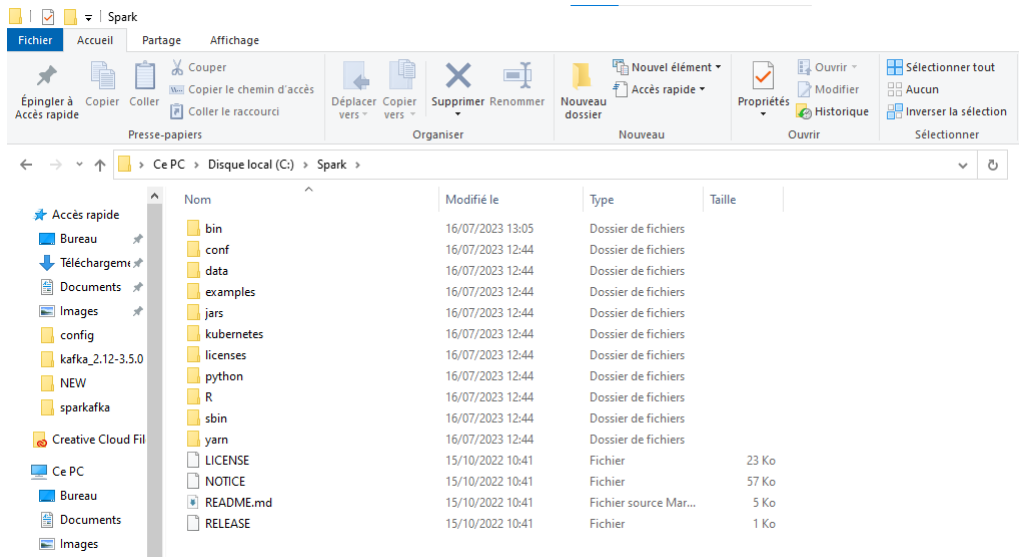
- Donc il faut créer les 2 répertoires `C :/kafka_2.12-3.5.0/data/kafka_data` et `C :/kafka_2.12-3.5.0/data/zookeeper`



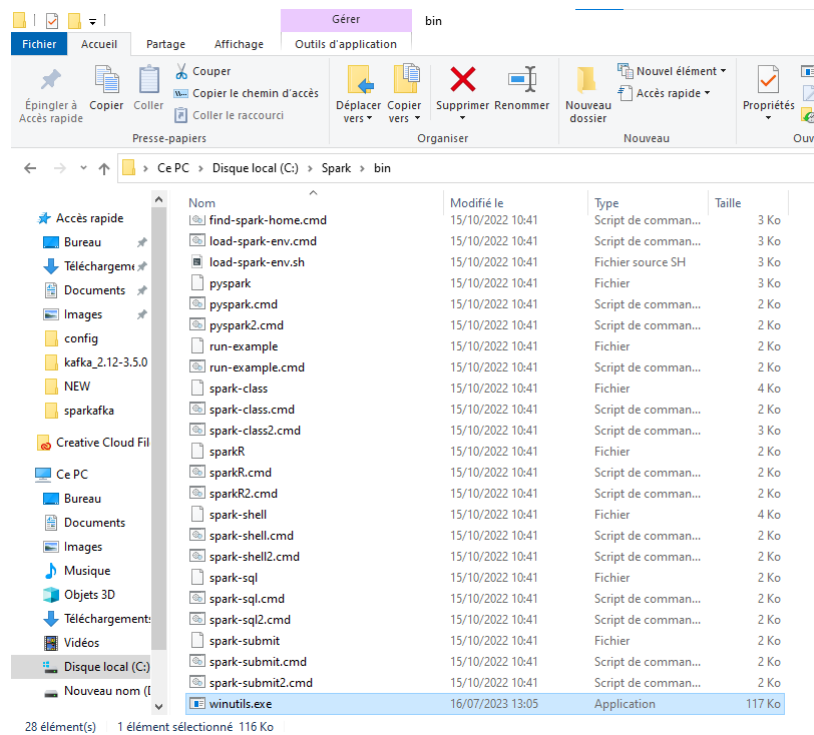
- l'étape suivante est de décompresser l'archive `spark-3.3.1-bin-hadoop2.tgz` dans le répertoire `C :/spark`



## Chapitre 1 Configuration d'événement

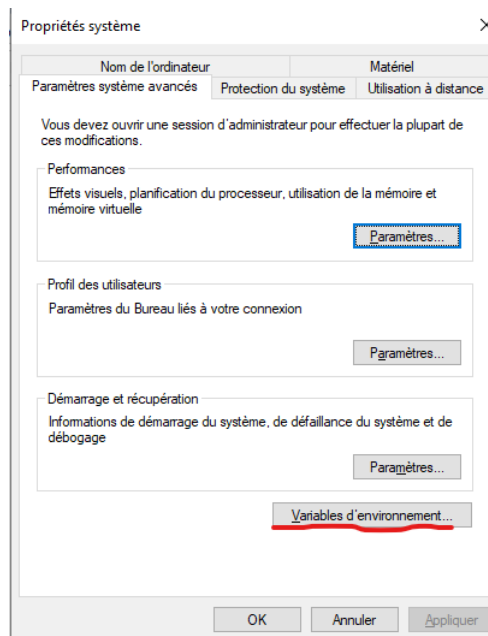
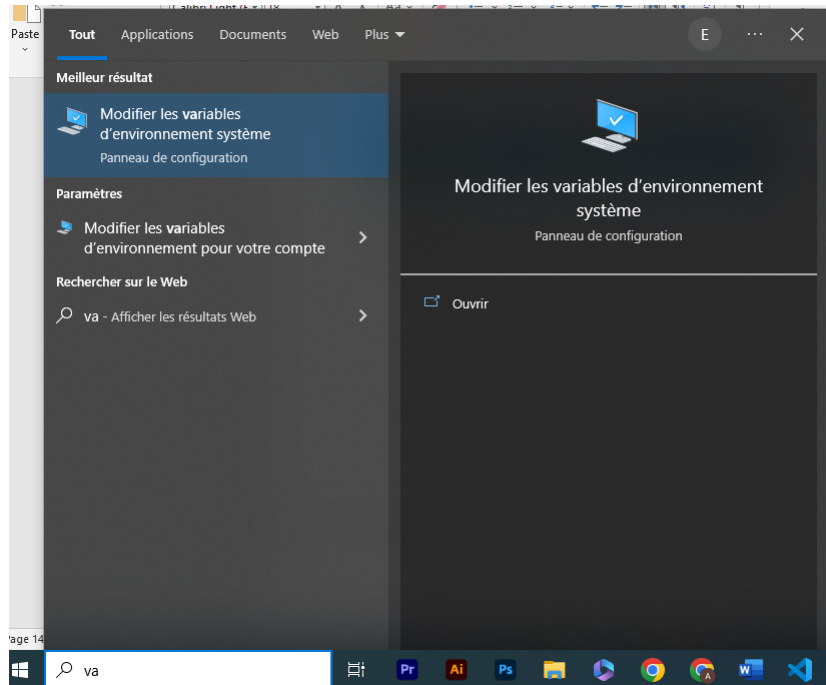


- Puis on va déplacer winutils.exe dans C :/spark/bin/



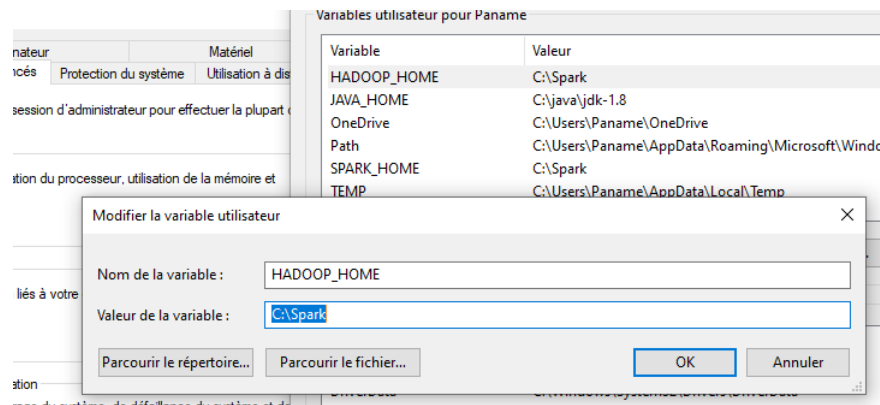
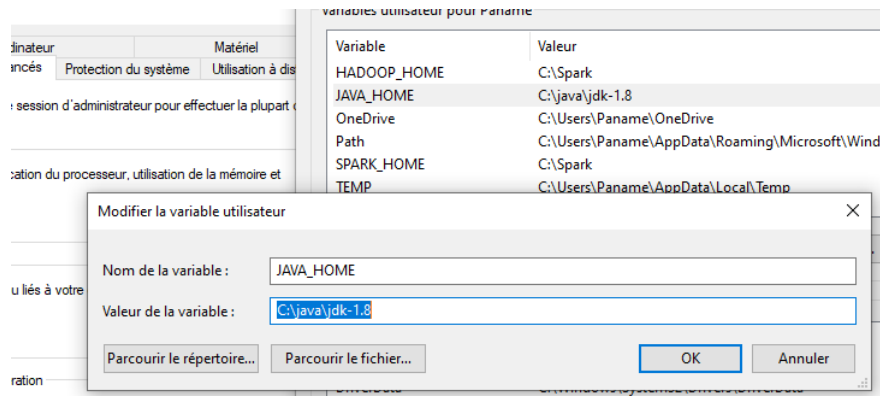
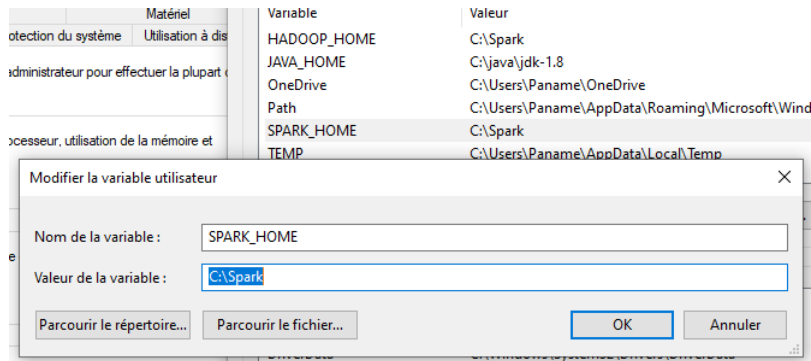
## 1.3 Modifications dans les variables d'environnement

- Dans la barre de recherche taper variable d'environnement



## Chapitre 1 Configuration d'événement

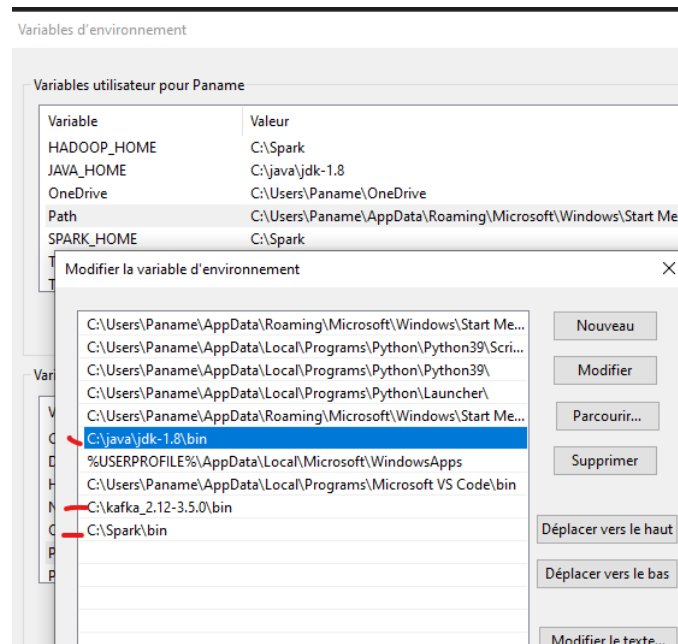
- Ajouter les paths suivants : SPARK\_HOME : C :/spark et HADOOP\_HOME : C :/spark et C :/java/jdk-1.8



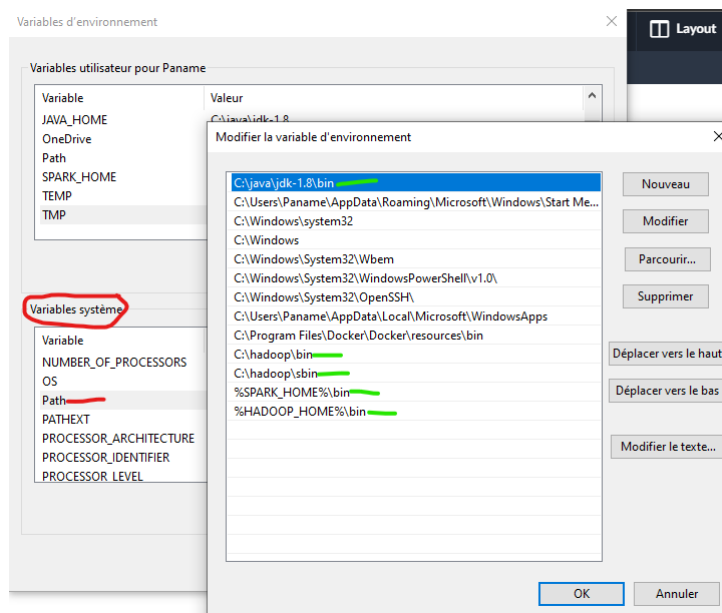


## Chapitre 1 Configuration d'événement

- Ajouter aussi les pathes suivants dans Path de utilisateur



- Ajouter aussi les pathes suivants dans path de système



## 1 Configuration de Docker Compose

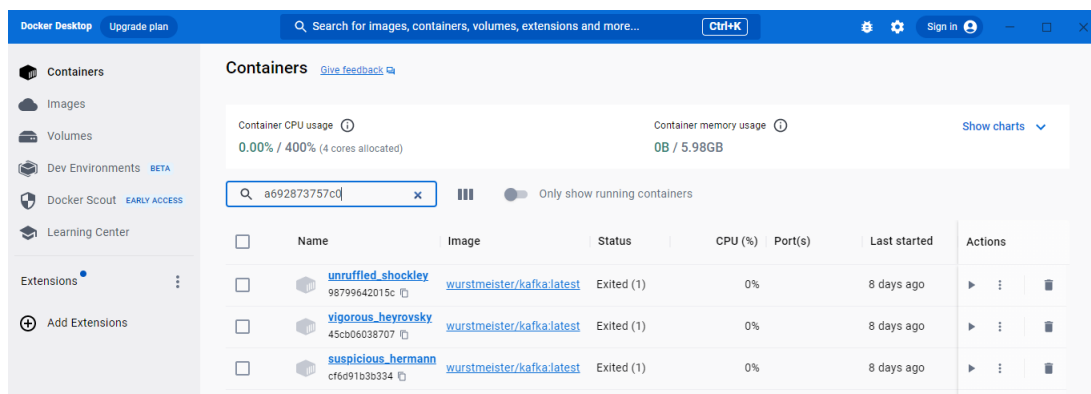
- Créer un fichier docker-compose.yml qui contient des images(containers) Zookeeper, Kafka server, Kibana et Elasticsearch

```
1 version: '3.7'
2
3 services:
4     zookeeper:
5         image: wurstmeister/zookeeper
6         ulimits:
7             nofile:
8                 soft: 65536
9                 hard: 65536
10        container_name: zookeeper
11        ports:
12            - "2181:2181"
13    kafka:
14        image: wurstmeister/kafka
15        container_name: kafka
16        ports:
17            - "9092:9092"
18        environment:
19            KAFKA_ADVERTISED_HOST_NAME: localhost
20            KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
21    elasticsearch:
22        image: docker.elastic.co/elasticsearch/elasticsearch:7.4.0
23        container_name: elasticsearch
24        restart: always
25        environment:
26            - xpack.security.enabled=false
27            - discovery.type=single-node
28        ulimits:
29            memlock:
30                soft: -1
31                hard: -1
32            nofile:
33                soft: 65536
34                hard: 65536
35        cap_add:
36            - IPC_LOCK
37        volumes:
38            - elasticsearch-data-volume:/usr/share/elasticsearch/data
39        ports:
40            - 9200:9200
```

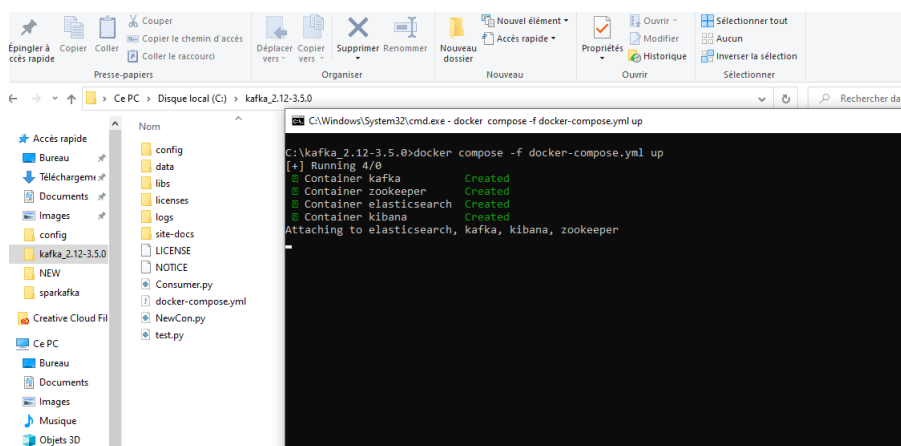
## Chapitre 1 Configuration d'événement

```
41     - 9300:9300
42
43     kibana:
44         container_name: kibana
45         image: docker.elastic.co/kibana/kibana:7.4.0
46         restart: always
47         environment:
48             - ELASTICSEARCH_HOSTS=http://elasticsearch:9200
49         ports:
50             - 5601:5601
51         depends_on:
52             - elasticsearch
53 volumes:
54     elasticsearch-data-volume:
55         driver: local
```

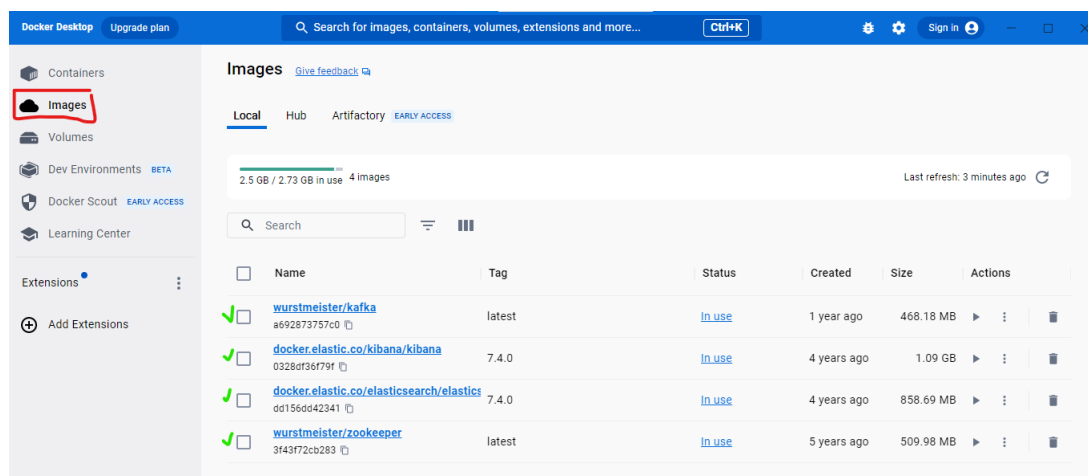
puis Lancer l'Application de Docker déjà installer



Et apres Taper la commande suivante



Finalement les containers seront bien installés



## 1.4 Kibana et Elasticsearch

### Elasticsearch

En quelques mots, nous aidons les utilisateurs à trouver plus rapidement ce qu'ils recherchent, qu'il s'agisse de collaborateurs ayant besoin de documents sur votre intranet ou de clients sur internet en quête de la paire de chaussures idéale. Pour aller un peu plus loin sur le plan technique, voici ce qu'on pourrait dire :

Elasticsearch est un moteur de recherche et d'analyse distribué gratuit et ouvert pour toutes les données (textuelles, numériques, géospatiales, structurées et non structurées). Elasticsearch a été conçu à partir d'Apache Lucene et a été lancé en 2010 par Elasticsearch N. V. (maintenant appelé Elastic). Réputé pour ses API REST simples, sa nature distribuée, sa vitesse et sa scalabilité, Elasticsearch est le composant principal de la Suite Elastic, un ensemble d'outils gratuits et ouverts d'ingestion de données, d'enrichissement, de stockage, d'analyse et de visualisation. Couramment appelée la Suite ELK (pour Elasticsearch, Logstash et Kibana), la Suite Elastic comprend désormais une riche collection d'agents de transfert légers, appelés les agents Beats, pour envoyer des données à Elasticsearch.

### Kibana

Kibana est une application frontend gratuite et ouverte qui s'appuie sur la Suite Elastic. Elle permet de rechercher et de visualiser les données indexées dans Elasticsearch. Si Kibana est connue pour être l'outil de représentation graphique de la Suite Elastic (précédemment appelée "la Suite ELK", acronyme d'Elasticsearch, Logstash et Kibana), elle sert aussi d'interface utilisateur pour le monitoring, la gestion et la sécurité des clusters de la Suite Elastic. Sans oublier qu'elle joue aussi le rôle de hub centralisé pour des solutions intégrées, développées sur

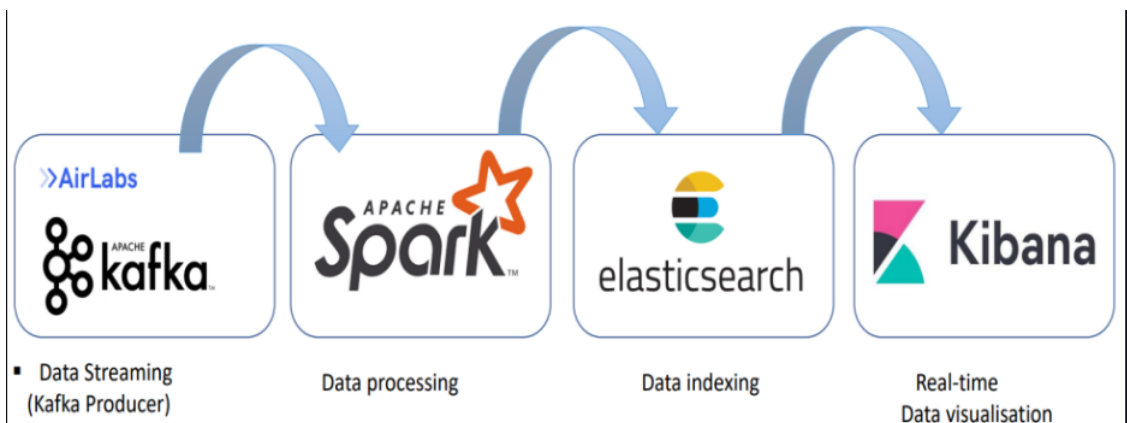
## *Chapitre 1 Configuration d'événement*

la Suite Elastic. Créée en 2013 au sein de la communauté Elasticsearch, Kibana s'est développée pour offrir une vue à 360° sur la Suite Elastic, devenant ainsi un véritable portail pour les utilisateurs et les entreprises.

# Chapitre 2

## Test d'Application

Dans ce projet, nous utiliserons une API de suivi de vols en temps réel, Apache Kafka, Elasticsearch et Kibana pour créer un pipeline de données d'informations de vol en temps réel et suivre les vols en direct. Nous utiliserons une architecture de haut niveau ainsi que les configurations correspondantes qui nous permettront de créer ce pipeline de données. Le résultat final sera un tableau de bord Kibana récupérant des données en temps réel à partir d'Elasticsearch.



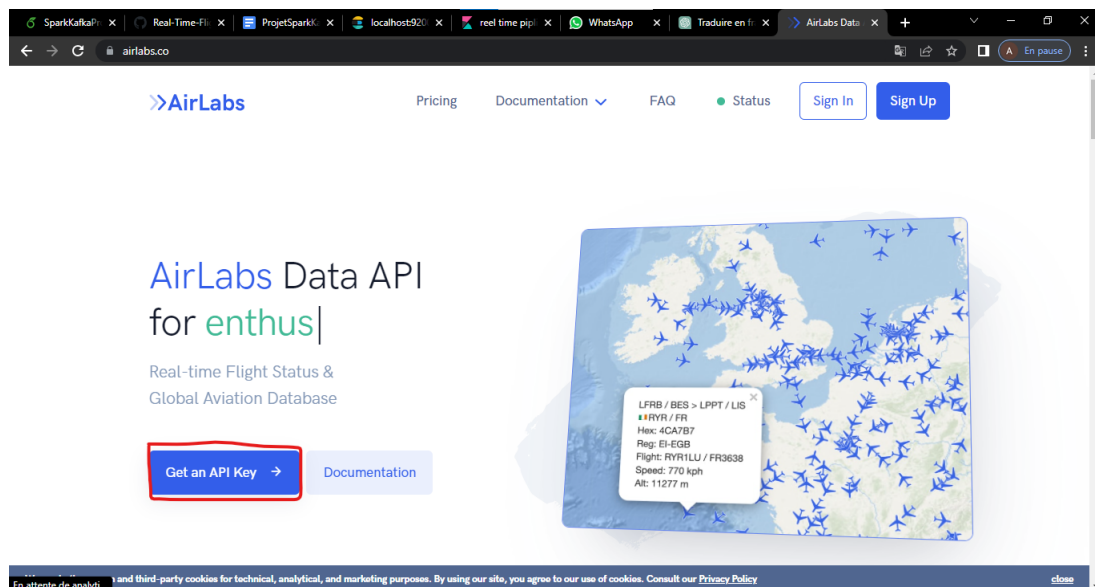
## Chapitre 2 Test d'Application

Nous avons commencé par collecter en temps réel des informations de vol "Flight Tracker API and Data - Aviation database and API"(numéro d'immatriculation de l'aéronef, latitude géographique de l'aéronef, longitude géographique de l'aéronef, élévation de l'aéronef, numéro de vol, etc.) et nous les avons ensuite envoyées à Kafka pour l'analyse.

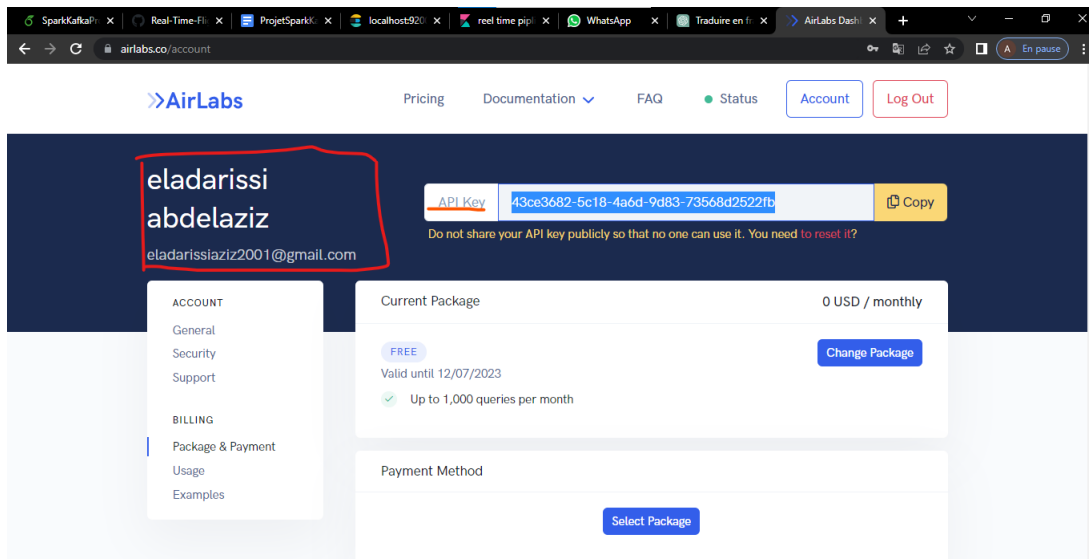
```
> $ curl https://airlabs.co/api/v9/flights

[{"hex": "76CCCD",
  "reg_number": "9V-SFM",
  "flight_iata": "SQ7371",
  "flight_icao": "SIA7371",
  "lat": 21.366613, "lng": 66.61321,
  "alt": 10021, "dir": 121,
  "speed": 966, "v_speed": -0.3,
  "airline_iata": "SQ", "airline_icao": "SIA",
  "dep_iata": "AMS", "dep_icao": "EHAM",
  "arr_iata": "SIN", "arr_icao": "WSSS",
  "flag": "SG", "aircraft_icao": "B744",
  "squawk": "3420", "updated": 1611366621
}]
```

il faut créer une compte dans ce site <https://airlabs.co/> pour avoir le KEY API et le clé secret



## Chapitre 2 Test d'Application



L'étape suivante consiste à créer un topic (test-topic) pour stocker les informations qui seront produites par le producteur.(Programme Java)

```
1 kafka-topics.bat --create --topic test-topic --bootstrap-server localhost:9092 --partitions 2 --replication-factor 2
```

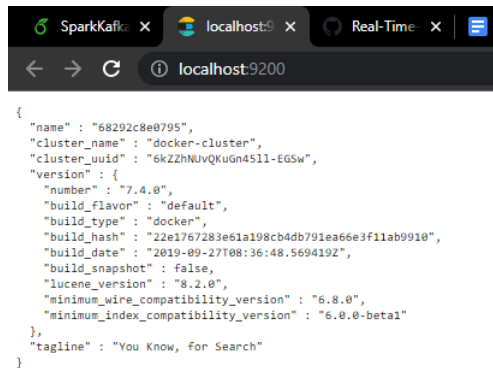
Puis Démarrer les serveurs kafka,zookeeper,Kibana et ElasticSearch

```
C:\Windows\System32\cmd.exe - docker compose up
C:\kafka_2.12-3.5.0>docker compose up
[+] Running 4/0
  Container elasticsearch Created 0.0s
  Container zookeeper Created 0.0s
  Container kafka Created 0.0s
  Container kibana Created 0.0s
Attaching to elasticsearch, kafka, kibana, zookeeper
kafka | [Configuring] 'port' in '/opt/kafka/config/server.properties'
kafka | [Configuring] 'advertised.host.name' in '/opt/kafka/config/server.properties'
kafka | Excluding KAFKA_HOME from broker config
zookeeper | ZooKeeper JMX enabled by default
zookeeper | Using config: /opt/zookeeper-3.4.13/bin/./conf/zoo.cfg
kafka | [Configuring] 'log.dirs' in '/opt/kafka/config/server.properties'
kafka | Excluding KAFKA_VERSION from broker config
kafka | [Configuring] 'zookeeper.connect' in '/opt/kafka/config/server.properties'
kafka | [Configuring] 'broker.id' in '/opt/kafka/config/server.properties'
zookeeper | 2023-07-21 22:10:49,913 [myid:] - INFO [main:QuorumPeerConfig@136] - Reading configuration from: /opt/zookeeper-3.4.13/bin/./conf/zoo.cfg
zookeeper | 2023-07-21 22:10:49,925 [myid:] - INFO [main:DatadirCleanupManager@78] - autopurge.snapRetainCount set to 3
zookeeper | 2023-07-21 22:10:49,925 [myid:] - INFO [main:DatadirCleanupManager@79] - autopurge.purgeInterval set to 1
zookeeper | 2023-07-21 22:10:49,928 [myid:] - WARN [main:QuorumPeerMain@116] - Either no config or no quorum defined in config, running in standalone mode
zookeeper | 2023-07-21 22:10:49,928 [myid:] - INFO [PurgeTask:DatadirCleanupManager$PurgeTask@138] - Purge task started.
zookeeper | 2023-07-21 22:10:49,956 [myid:] - INFO [PurgeTask:DatadirCleanupManager$PurgeTask@144] - Purge task completed.
zookeeper | 2023-07-21 22:10:49,976 [myid:] - INFO [main:QuorumPeerConfig@136] - Reading configuration from: /opt/zookeeper-3.4.13/bin/./conf/zoo.cfg
zookeeper | 2023-07-21 22:10:49,990 [myid:] - INFO [main:ZooKeeperServerMain@98] - Starting server
zookeeper | 2023-07-21 22:10:50,012 [myid:] - INFO [main:Environment@100] - Server environment:zookeeper.version=3.4.13-2d71af4dbe22557fda74f9a9b4309b15a7487f03,
built on 06/29/2018 04:05 GMT
zookeeper | 2023-07-21 22:10:50,012 [myid:] - INFO [main:Environment@100] - Server environment:host.name=6264158797fd
zookeeper | 2023-07-21 22:10:50,013 [myid:] - INFO [main:Environment@100] - Server environment:java.version=1.7.0_65
zookeeper | 2023-07-21 22:10:50,015 [myid:] - INFO [main:Environment@100] - Server environment:java.vendor=Oracle Corporation
zookeeper | 2023-07-21 22:10:50,015 [myid:] - INFO [main:Environment@100] - Server environment:java.home=/usr/lib/jvm/java-7-openjdk-amd64/jre
zookeeper | 2023-07-21 22:10:50,016 [myid:] - INFO [main:Environment@100] - Server environment:java.class.path=/opt/zookeeper-3.4.13/bin/./build/classes:/opt/zoo
keeper-3.4.13/bin/./build/lib/*:jar:/opt/zookeeper-3.4.13/bin/./lib/slf4j-log4j12-1.7.25.jar:/opt/zookeeper-3.4.13/bin/./lib/slf4j-api-1.7.25.jar:/opt/zookeeper-3.4.
13/bin/./lib/netty-3.10.6.Final.jar:/opt/zookeeper-3.4.13/bin/./lib/log4j-1.2.17.jar:/opt/zookeeper-3.4.13/bin/./lib/jline-0.9.94.jar:/opt/zookeeper-3.4.13/bin/./li
b/audience-annotations-0.5.0.jar:/opt/zookeeper-3.4.13/bin/./zookeeper-3.4.13.jar:/opt/zookeeper-3.4.13/bin/./src/java/lib/*:jar:/opt/zookeeper-3.4.13/bin/./conf:
```

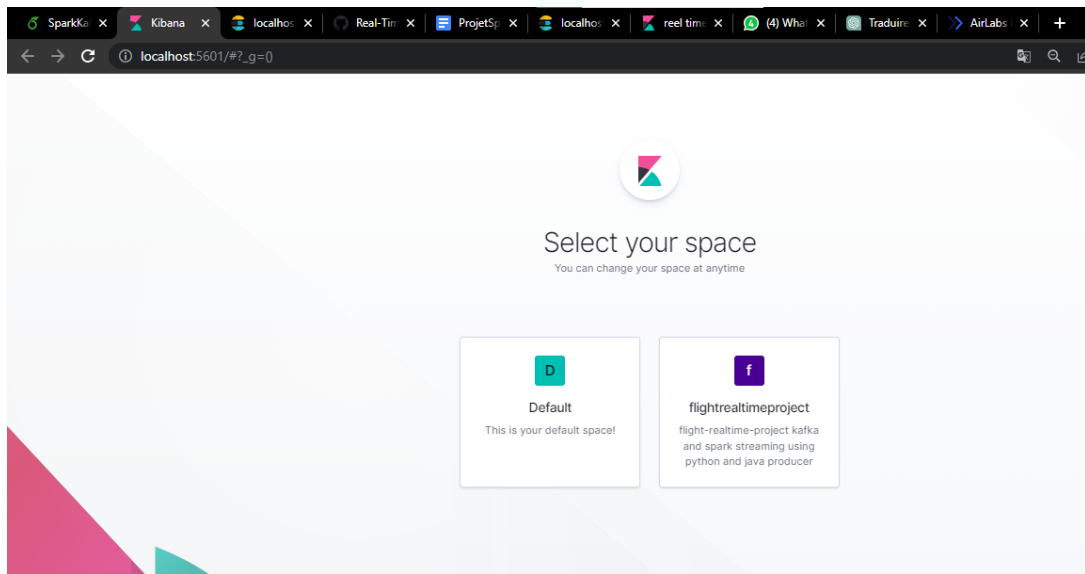


## Chapitre 2 Test d'Application

Pour ElasticSearch Vous pouvez accéder à <http://localhost:9200> pour vérifier s'il est actif et opérationnel. et Kibana <http://localhost:5601>

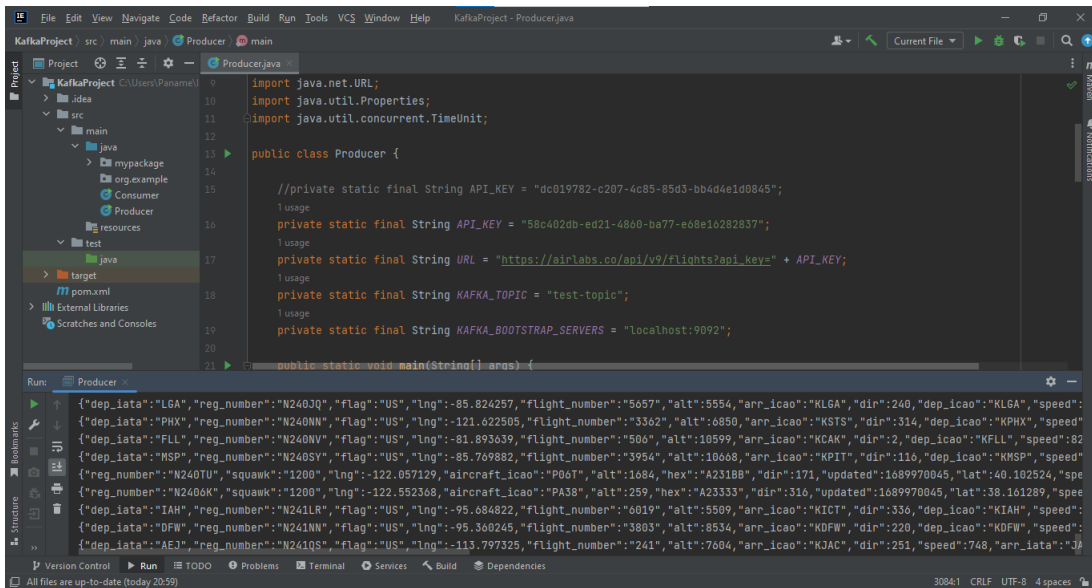


```
{
  "name" : "68292c8e0795",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "6kZZhNuvQKuGn4511-EGSw",
  "version" : {
    "number" : "7.4.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "22e1767283e61a198cb4db791ea66e3f11ab9910",
    "build_date" : "2019-09-27T08:36:48.569419Z",
    "build_snapshot" : false,
    "lucene_version" : "8.2.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```



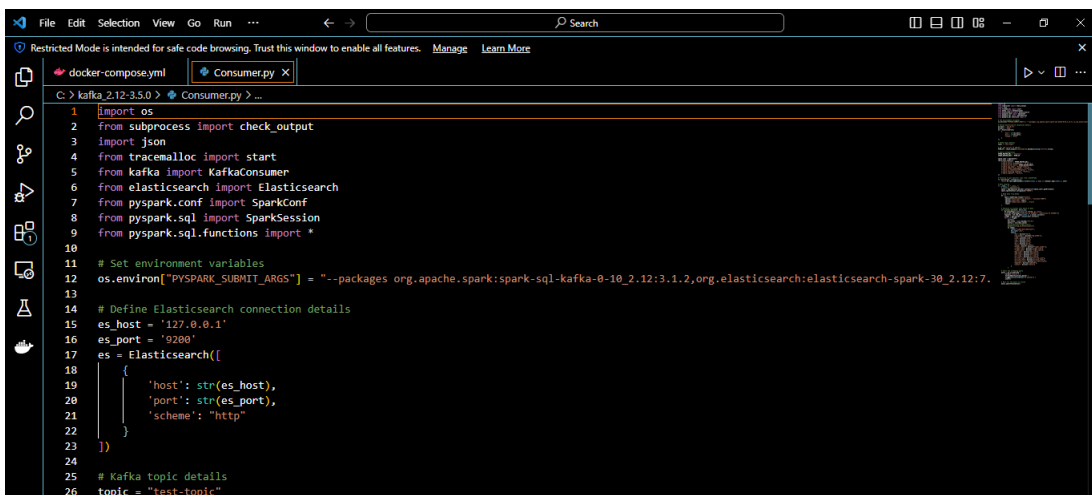
## Chapitre 2 Test d'Application

Les données sont récupérées à partir de l'API de flux de données de vol et envoyées vers un topic Kafka à l'aide de Programme Java(Producer.java) qui Nous avons créer.



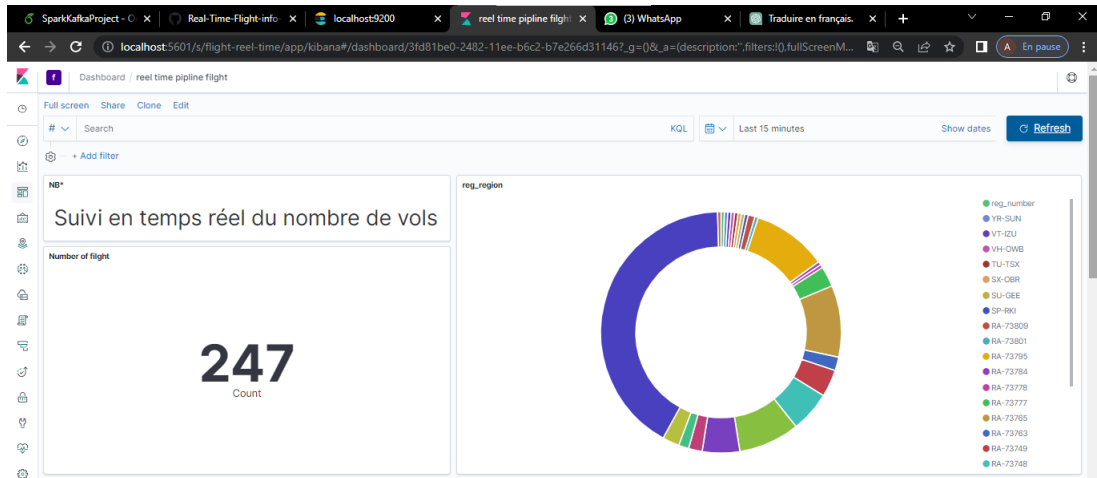
En fin On va démarrer le programme Python (Consumer.py) pour collecter périodiquement des données en temps réel à partir du topic Kafka et les envoyer dans un index Elasticsearch. EN Utilisant la commande suivante

```
spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.1.2,
org.elasticsearch:elasticsearch-spark-30_2.12:7.14.2 Consumer.py
```



## Chapitre 2 Test d'Application

### Dashboard Final



The table displays flight information for 247 flights. The columns are: reg\_number, hex, \_id, status, arr\_lata, dep\_lata, airline\_lata, alt, dir, lat, lng, and speed. The table is paginated, showing 101-150 of 247 flights.

reg_number	hex	_id	status	arr_lata	dep_lata	airline_lata	alt	dir	lat	lng	speed
RA-09607	142587	wrggkBCOV74R0XUqj	en-route	LED	RGK	4G	12192	258	56.545029	46.745495	833
RA-09607	142587	xLgglkBCOV74RXYErg	en-route	LED	RGK	4G	12192	258	56.545029	46.745495	833
RA-73734	152006	xbgfkakBCOV74R0Xk4	en-route	OMS	SVO	SU	11285	83	56.050106	43.6619	905
RA-73734	152006	xrgLackBCOV74R0XU9	en-route	OMS	SVO	SU	11285	83	56.050106	43.6619	905
RA-73734	152006	yLgMeokBCOV74R0XUo9	en-route	OMS	SVO	SU	11285	83	56.050106	43.6619	905
RA-73734	152006	ynlgMeokBCOV74R0XUpe	en-route	OMS	SVO	SU	11285	83	56.050106	43.6619	905
N331DN	A390DE	x7gMeokBCOV74R0Xk6o	en-route	ATL	LGA	DL	10332	237	39.710632	-76.563957	740
SU-GEE	010163	w7gggkBCOV74R0XOExpr	en-route	BRU	CAI	MS	11582	334	42.811984	22.852872	772
NS04NN	AC7F41	ybgMeokBCOV74R0XYE13	en-route	DFW	DFW	AA	2819	24	33.167084	-96.97599	546
OE-LNX	440BCE	07gMeokBCOV74R0XUq2	en-route	EMA	DUB	QV	6393	104	53.364487	-6.070025	750
RA-73734	152006	tLgDeokBCOV74R0XUuN	en-route	OMS	SVO	SU	11277	86	56.110249	44.633076	894
RA-73734	152006	z7gMeokBCOV74R0XZUu	en-route	OMS	SVO	SU	11269	83	56.096287	44.387512	896
RA-73734	152006	OLgMeokBCOV74R0XkqF	en-route	OMS	SVO	SU	11269	83	56.101265	44.469223	896
RA-73734	152006	0hgMeokBCOV74R0X4E5	en-route	OMS	SVO	SU	11277	86	56.110249	44.633076	894
RA-73734	152006	zLgMeokBCOV74R0XkzPZ	en-route	OMS	SVO	SU	11269	83	56.096287	44.387512	896
RA-73734	152006	zrghMeokBCOV74R0XFQjB	en-route	OMS	SVO	SU	11269	83	56.096287	44.387512	896
N912WN	ACA01A	y7gMeokBCOV74R0XqoH	en-route	DEN	BNA	WN	10736	269	38.844955	-102.205383	764

