*Université Chouaib Doukkali*

*Ecole Nationale des Sciences Appliquées d’El Jadida*

*Département Télécommunications, Réseaux et Informatique*

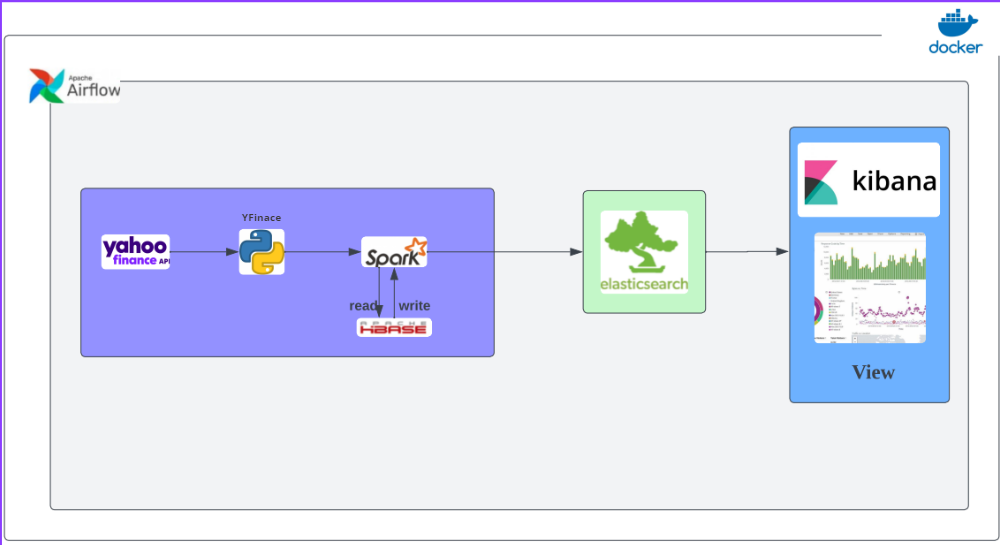
***Filiere : 2ITE***

***Niveau : 3 eme Année***

***Sujet :***

***Automatisation d’un pipeline Big Data de visualisation***

***de données en utilisant Apache Airflow et ELK Stack***

******

***Réalisé Par : Encadré par :***

* ***Hmayda Abdessamad - Pr. Fahd Kalloubi***
* ***KADIRI Mariem***

Table des matières

[**I.** **Objectif du projet :** 2](#_Toc154604415)

[***II.*** **Prérequis : Installation et Configuration** 2](#_Toc154604416)

[**III.** **Architecture générale et workflow :** 3](#_Toc154604417)

[**IV.** **Création d’un DAG :** 5](#_Toc154604418)

[**V.** **L’extraction des données à partir de Yahoo et stockage dans HBase** 7](#_Toc154604419)

[**VI.** **De Hbase vers Elasticsearch :** 9](#_Toc154604420)

[**VII.** **Visualisation des données dans Kibana** 10](#_Toc154604421)

# **Objectif du projet :**

Ce projet vise à automatiser un pipeline Big Data pour la collecte, le traitement et la visualisation de données financières. A partir de l’API Yahoo Finance, en utilisant la bibliothèque yfinance pour l’extraction des données, Apache Airflow pour l’automatisation du pipeline, HBase pour le stockage initial, et Spark SQL pour l'accès et le traitement des données. Des indicateurs techniques sont ajoutés, et les données traitées sont stockées dans Elasticsearch. La visualisation est réalisée via Kibana, avec l'accès à Elasticsearch à travers Spark, permettant la création des visualisations personnalisées pour une analyse approfondie des données financières.

# **Prérequis : Installation et Configuration**

Ce projet nécessite l'installation et la configuration préalable de plusieurs outils et environnements. Suivez attentivement ces étapes pour exécuter ce projet chez vous.

## **Environnement de Développement :**

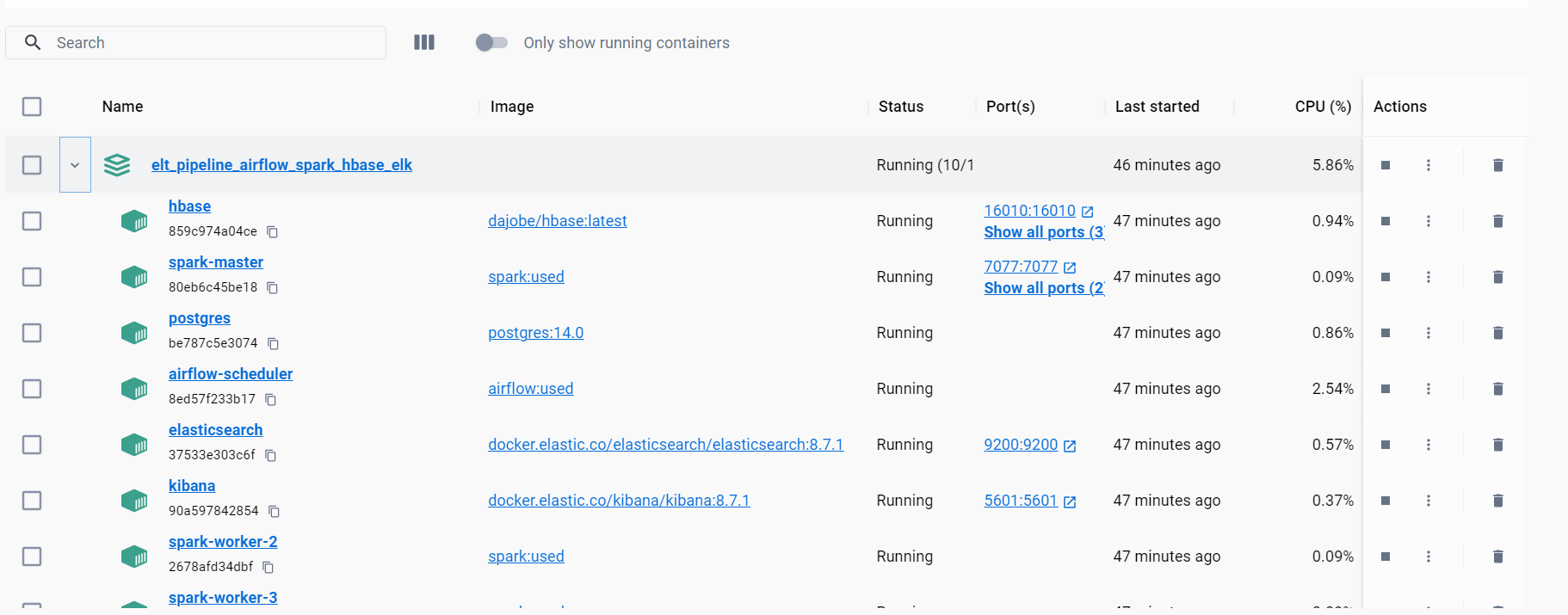
* Système d'Exploitation : Vérifiez que vous disposez d'un système d'exploitation compatible avec les outils nécessaires (Exemple : Ici, on utilise Windows).
* RAM : Assurez-vous d'avoir une mémoire RAM supérieure à 13 Go.
* Docker :
  + Téléchargez et installez Docker en suivant les instructions spécifiques à votre système d'exploitation : lien vers Docker
  + Vérifiez l'installation avec la commande : docker –version

## **Exécution du projet :**

* Tout d’abord allez cloner le projet

<https://github.com/Hmayda-Abdessamad/ELT_pipeline_Airflow_Spark_Hbase_ELK.git>

* Accéder au dossier du projet et tapez docker compose up
  + Au début le OS essaye de  ‘puller’ les images docker utilisés dans ce projet
  + Après 10 conteneurs seront lancés

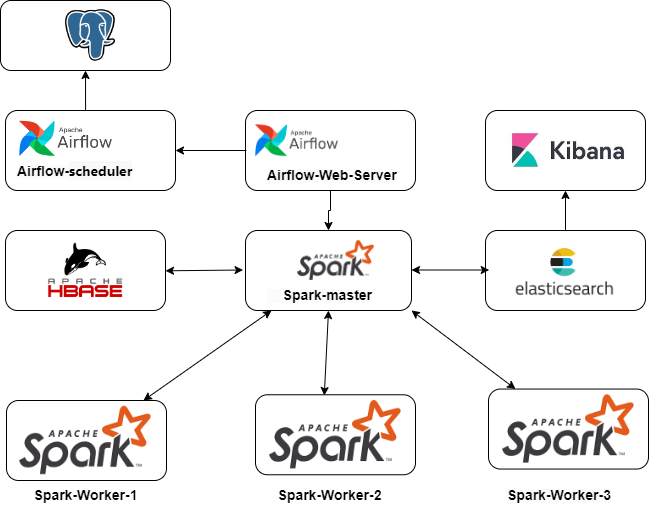


Vous pouvez maintenant accéder à airflow web server et lancez les différents dags pour qu’ils puissent s’exécuter

# **Architecture générale et workflow :**

## **Architecture générale**

Comme on a mentionné précédent, lors de l’exécution du projet, notre cluster comportera 10 conteneurs, qui sont les suivants :



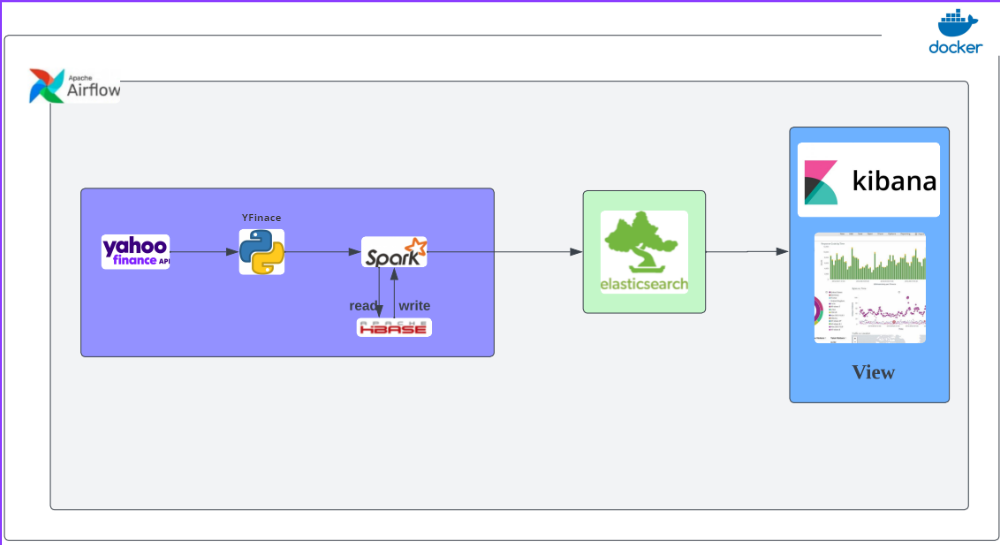
Apres l’exécution du projet vous pouvez accéder aux services dans un navigateur web, utilisez les URL suivantes :

* HBase : <http://localhost:16010>
* Spark : <http://localhost:8081>
* Airflow : <http://localhost:8080>
* Kibana : <http://localhost:5601>

Assurez-vous que les services sont correctement démarrés et que les ports spécifiés dans votre fichier docker-compose.yml ne sont pas utilisés par d'autres applications sur votre système

## **Workflow :**

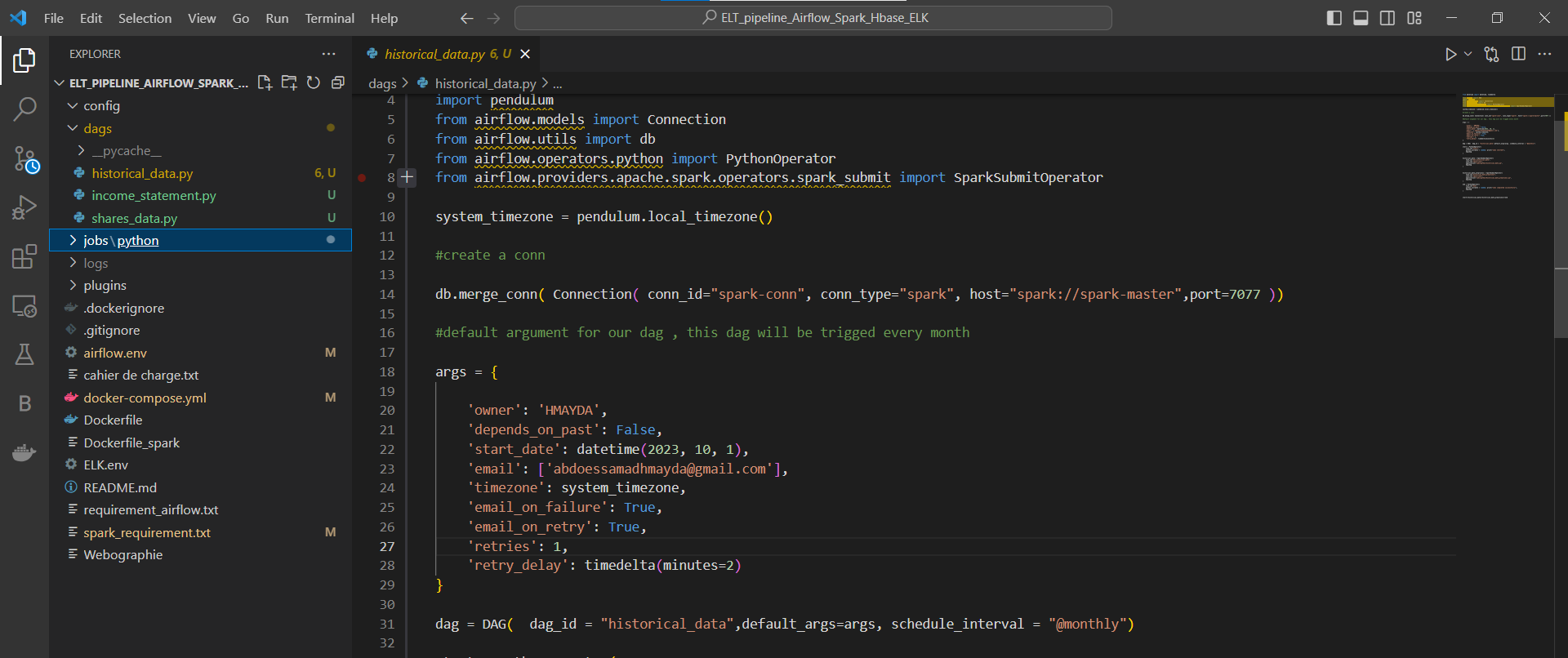
La figure suivante décrit les différentes étapes de notre pipeline



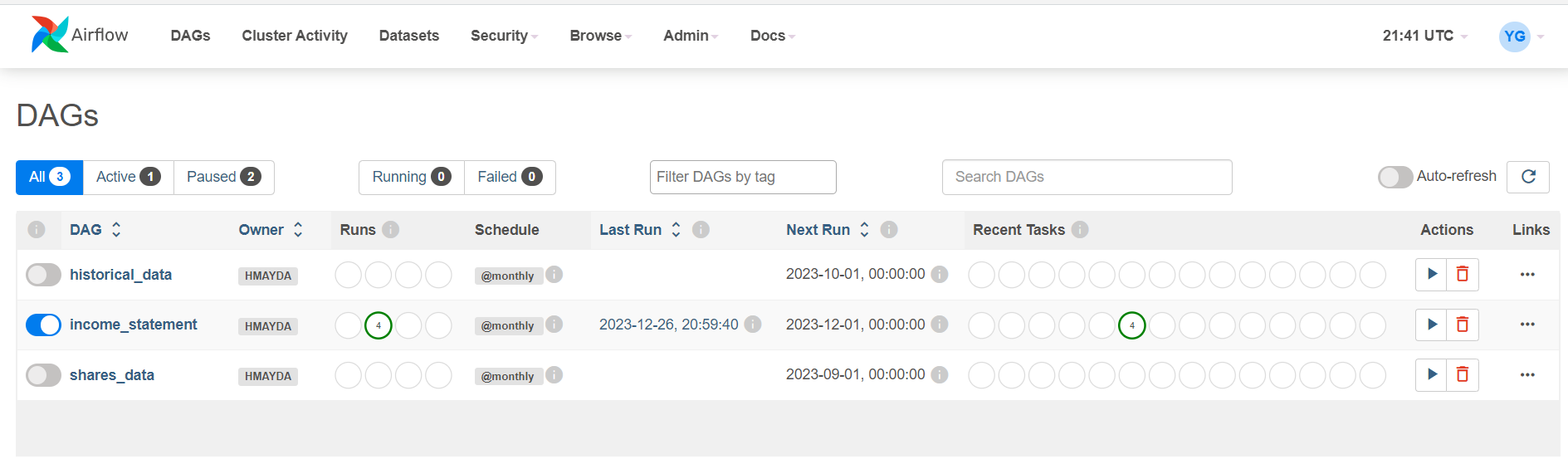
* Au début, des données de type variés sont extraire de l’API Yahoo finance, grâce à la bibliothèque yfinance <https://pypi.org/project/yfinance/>
* Des traitements basics (ex : nettoyage, jointure) seront exercé sur ces données, ces traitements sont faits avec pyspark et pyspark.sql
* Les données résultantes de ce traitement seront par la suite stockées dans HBase
* Encore une fois les données, on retire les données stockées dans HBase on calcule d’autre métriques (ex : max, médian …) Et les données résultantes seront stockées dans Elasticsearch
* Vers la fin on peut visualiser les données existe dans Elasticsearch sur Kibana

# **Création d’un DAG :**

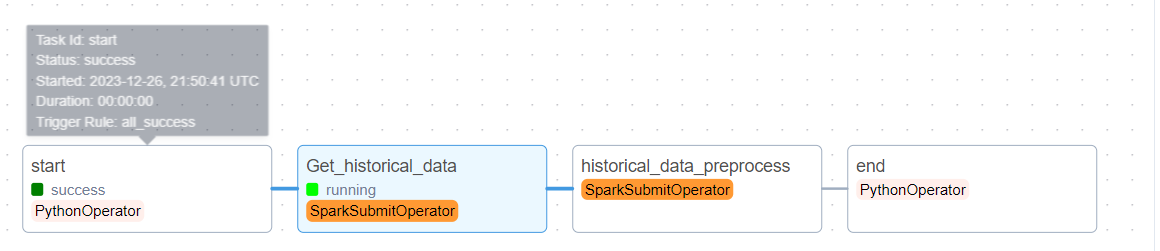
* Pour créer un DAG allez vers le dossier dags et ajouter un fichier python ( ex : historical\_data.py)
* la figure suivante montre l’exemple d’un DAG déjà créer , et ce dag sera exécuté chaque mois et à partir du 1/10/2023



* Pour voir les différents DAG qu’on a, on peut allez vers airflow-web-server <http://localhost:8080> ( username : admin , pwd : admin)



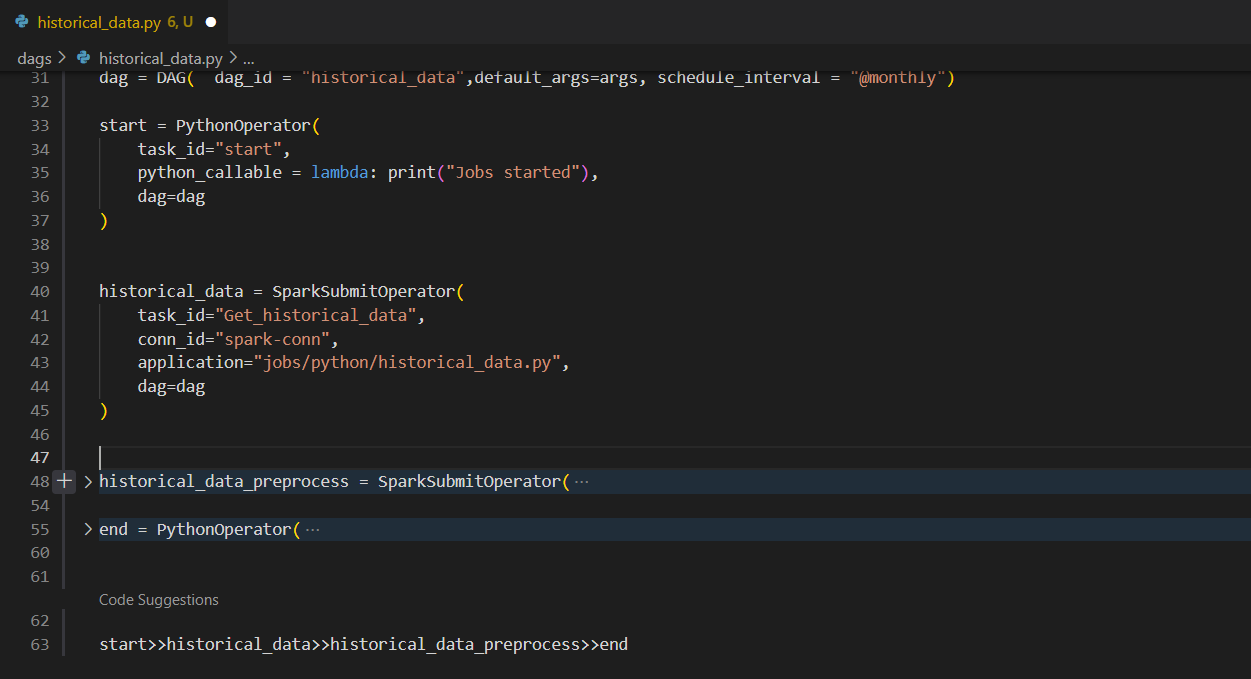
* On peut afficher chaque DAG sous forme de graphe, comme suivant :

******

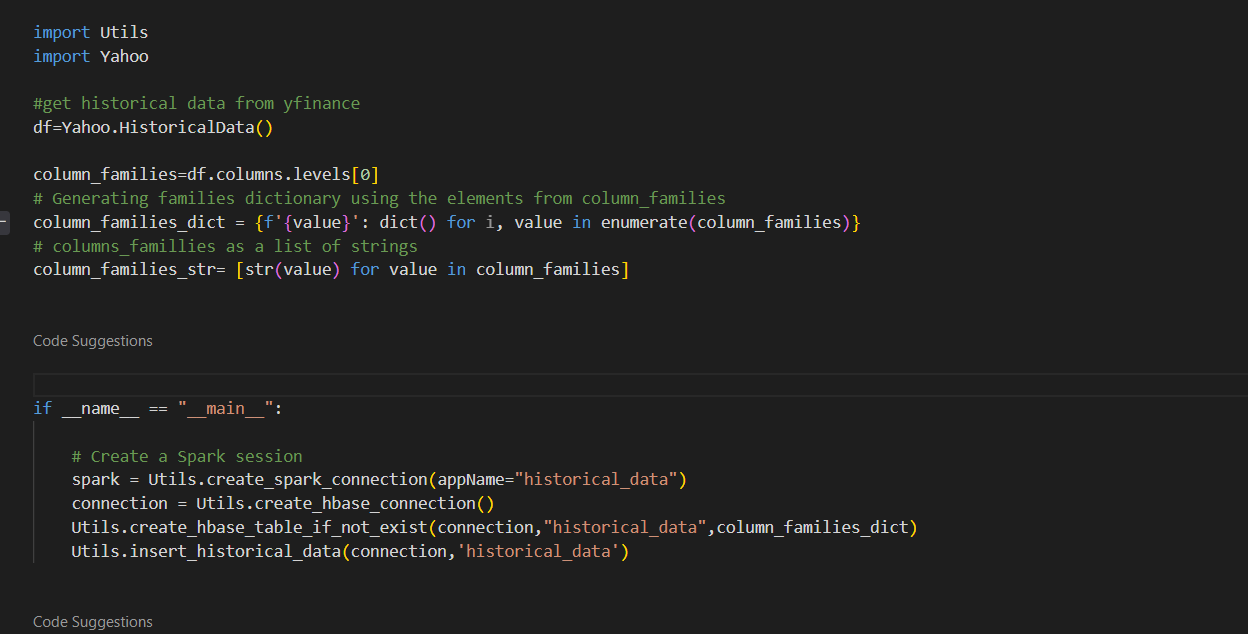
## **L’ajout d’une tache à un DAG**

Dans notre cas on a utilisé deux types d’operateur dans nos différentes taches au sein des différents dags

La figure suivante, décrit les taches existent dans le DAG « historical\_data » et leurs enchainements

******

# **L’extraction des données à partir de Yahoo et stockage dans HBase**

******

La figure précédente décrit ce que la deuxième tâche réalise dans notre DAG, donc au niveau de cette tâche on :

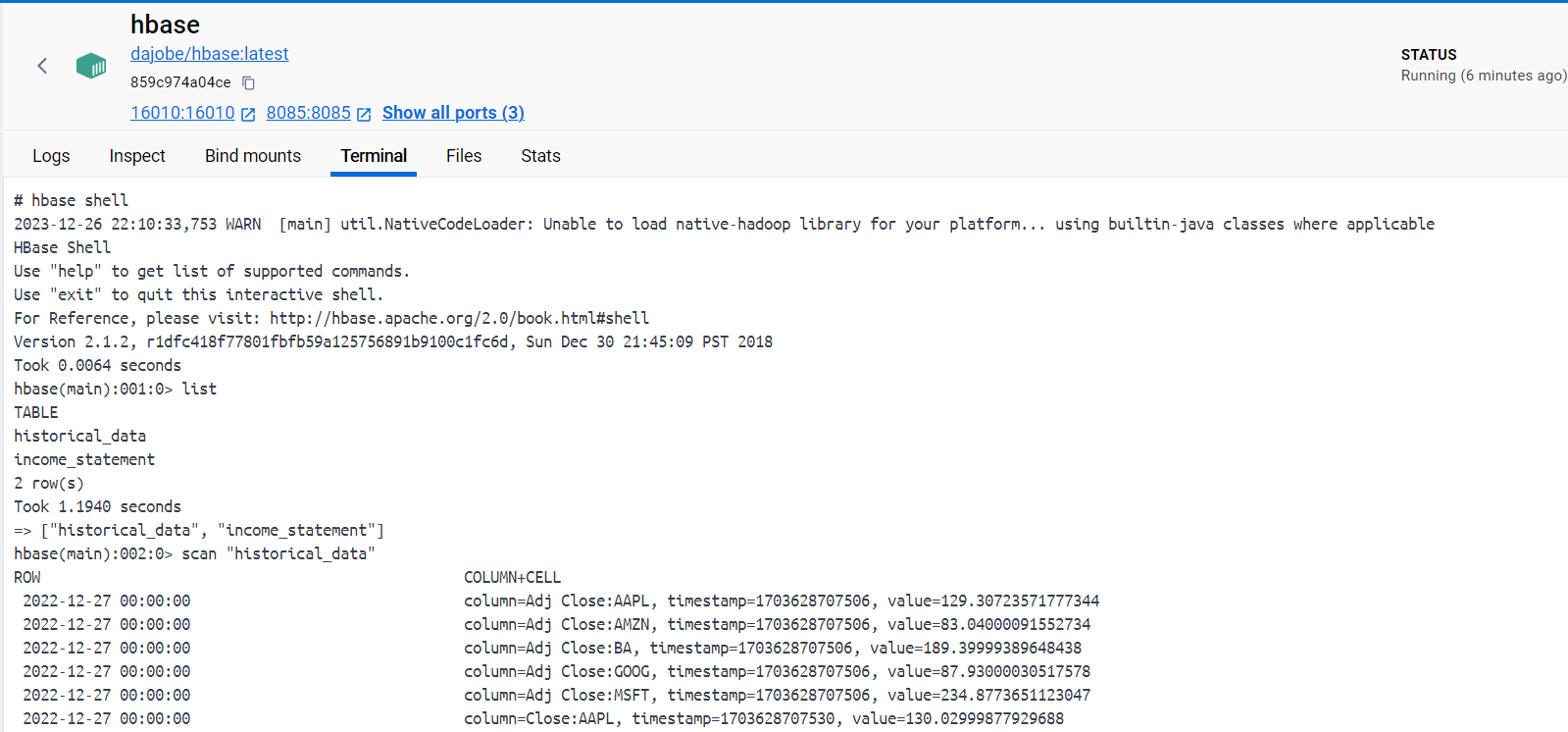
* Crée une application Spark
* Crée une connexion avec HBase

(En utilisant happybase <https://happybase.readthedocs.io/en/latest/index.html> )

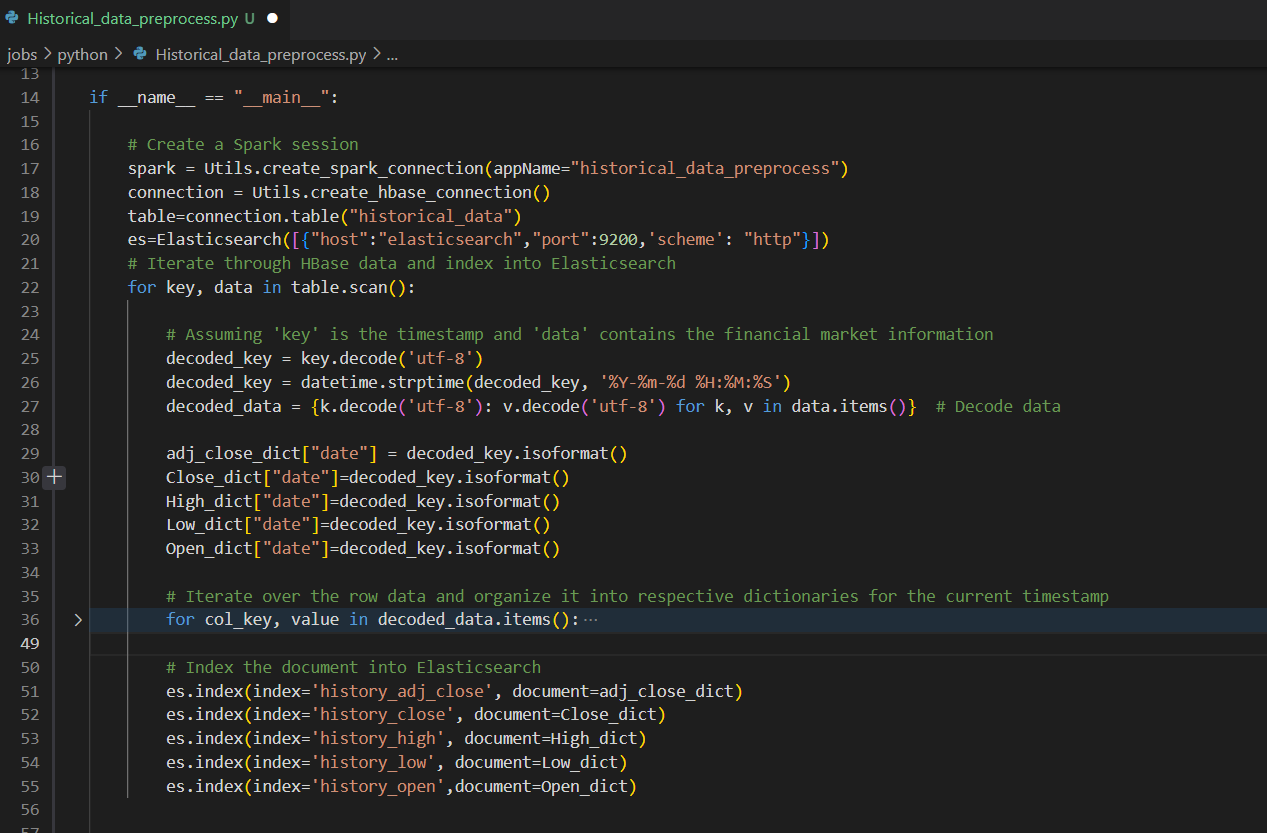
* Crée une table dans HBase
* Stock les donnees dans HBase

On peut, à chaque fois vérifiez que notre base donnée HBase contient les données juste apres l’exécution de la 2 eme tache de l’un des dag cités auparavant

Pour le faire, allez vers le bash de votre conteneur « hbase » et tapez « hbase shell » et vous pouvez interagir avec hbase comme suivant :



# **De Hbase vers Elasticsearch :**

******

La figure précédente décrit ce que la troisième tâche réalise dans notre DAG, donc au niveau de cette tâche on :

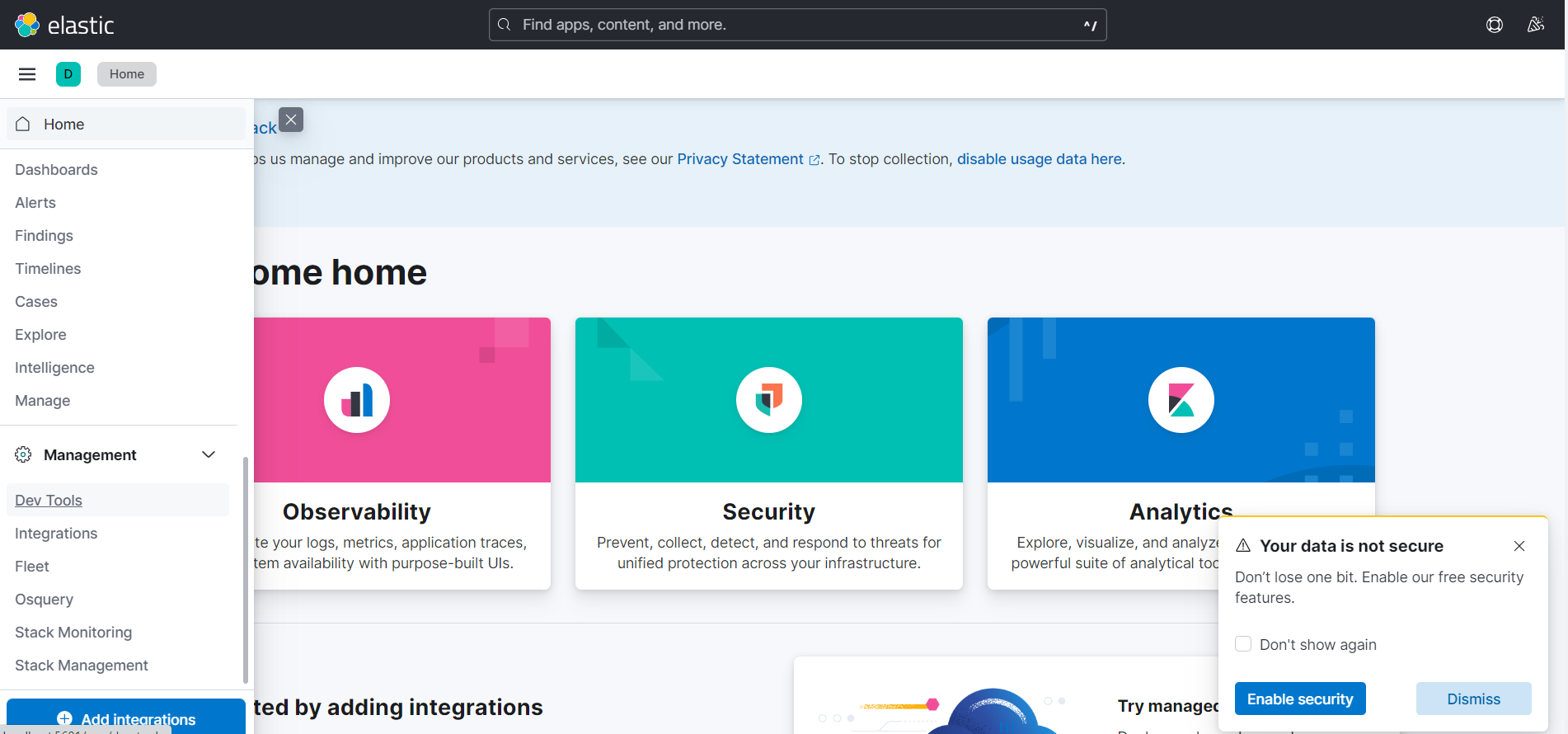
* Crée une application Spark
* Crée une connexion avec HBase

(En utilisant happybase <https://happybase.readthedocs.io/en/latest/index.html> )

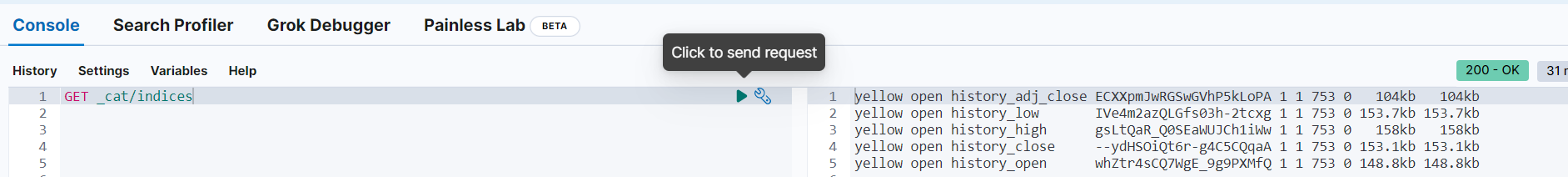
* Récupère une table dans HBase
* Etablie une connexion avec Elasticsearch (grâce à elasticsearch <https://pypi.org/project/elasticsearch/> )
* Boucle sur les enregistrements de Hbase on ajoute d’autre indicateur et on stock les donnees dans elasticsearch

🡺 On peut vérifiez que les données sont stockées dans elasticsearch

* Pour le faire allez vers kibana <http://localhost:5601> et allez vers Dev Tools



* Au niveau de la console, on peut afficher les indexes créées

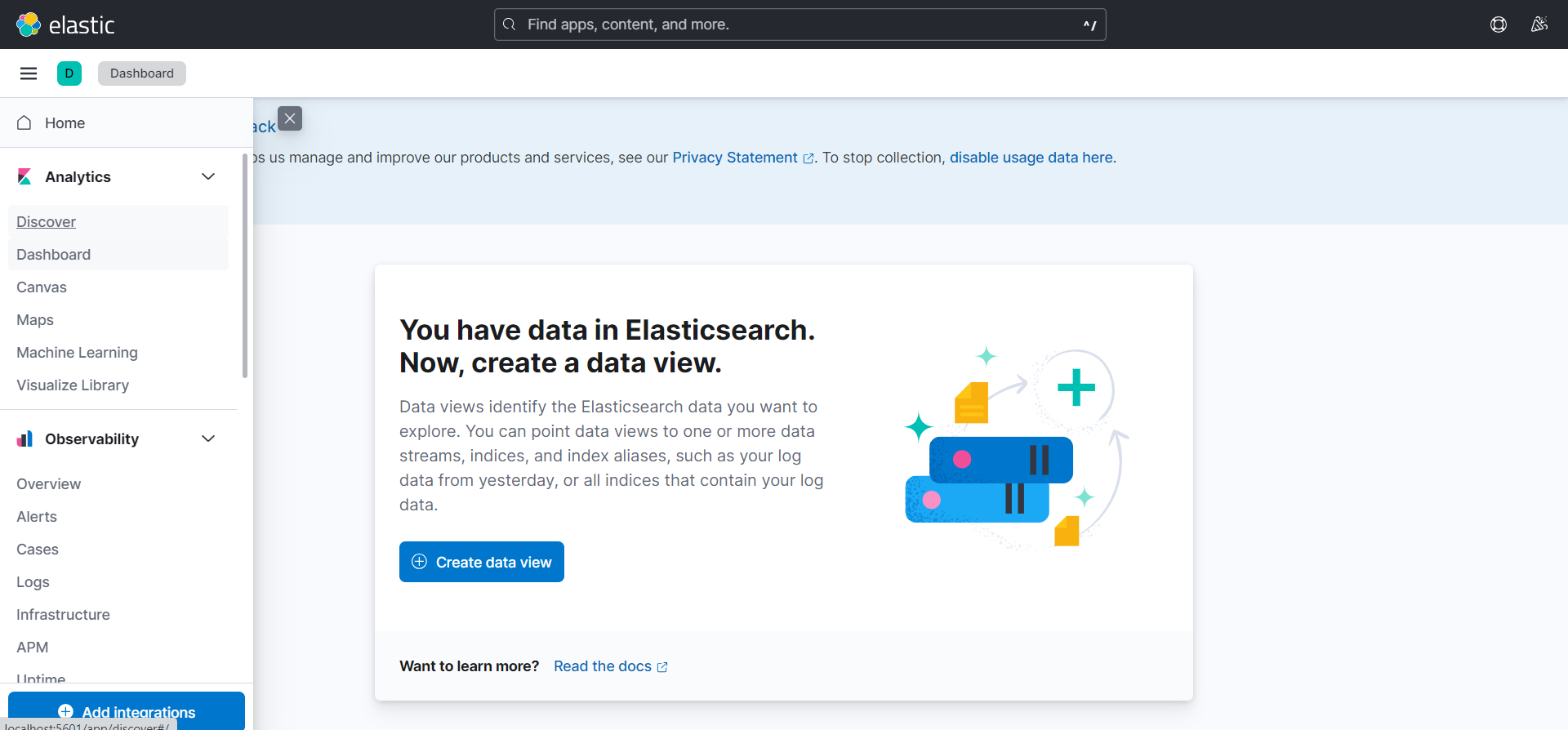


# **Visualisation des données dans Kibana**

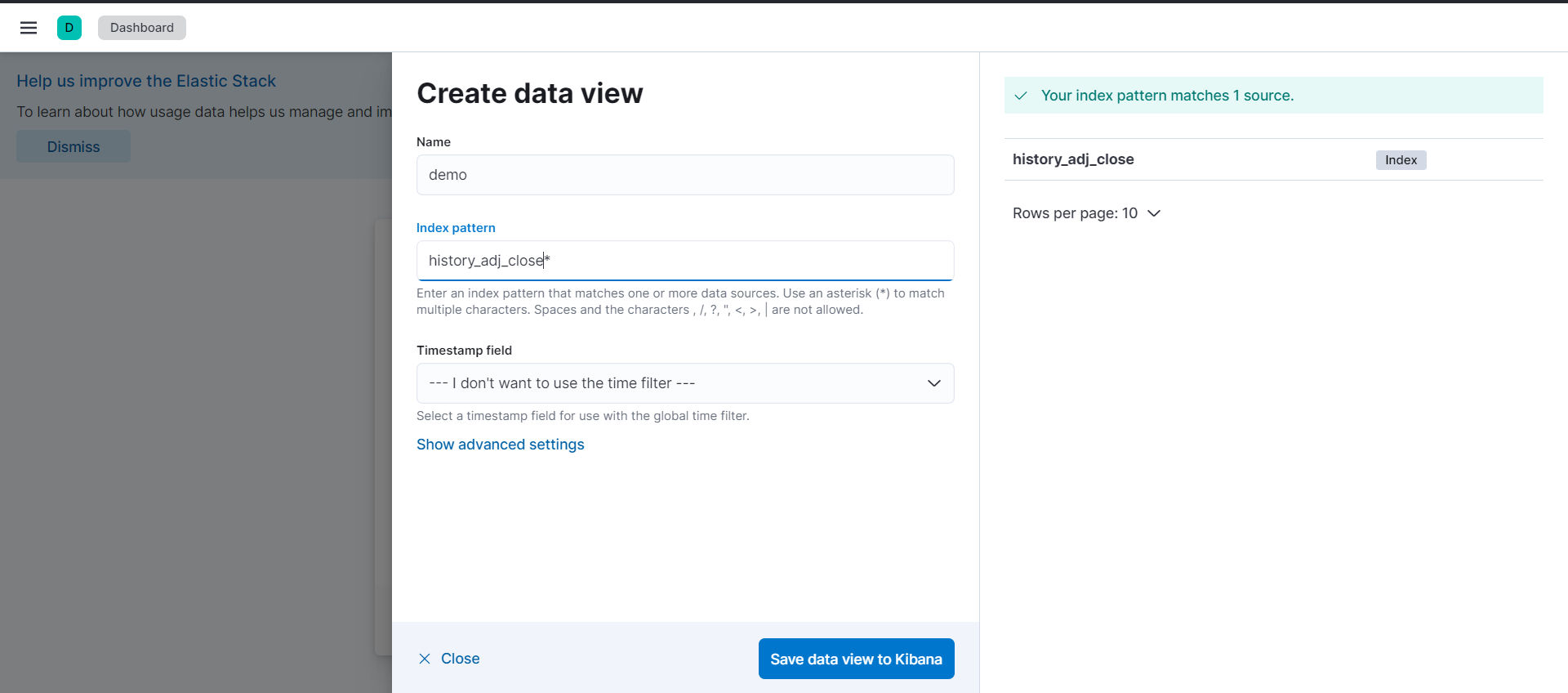
Kibana nous permet de créer des visualisation et Dashboard personnalisés

Essayons de créer notre première visualisation :

* Allez vers **Analytics >> Dashboard**

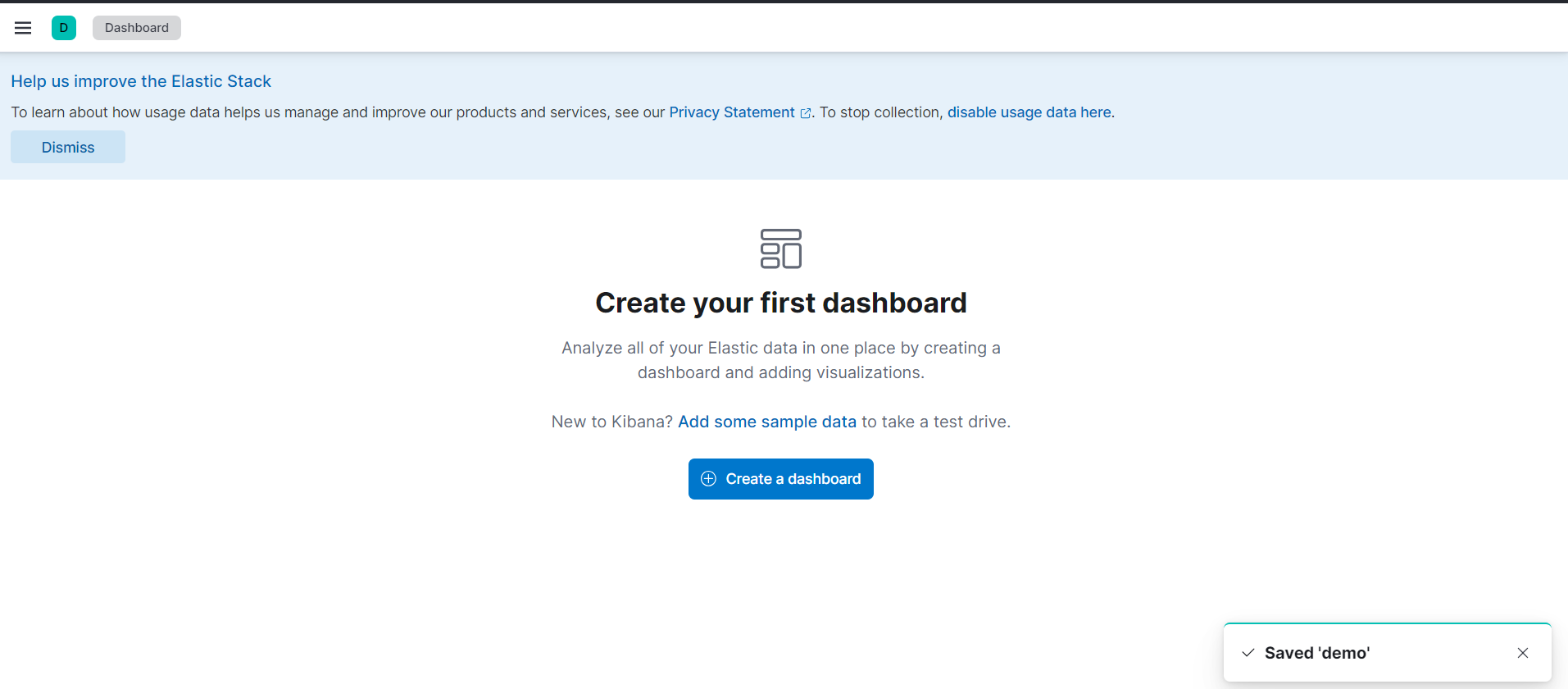


* Donc on doit tout d’abord créer une vue de données fait référence aux données extraites des indices Elasticsearch, pour le faire cliquez sur **Create data view**
* La fenêtre suivante s’affiche , on doit remplir les champs requises et enregistrer la vue de données

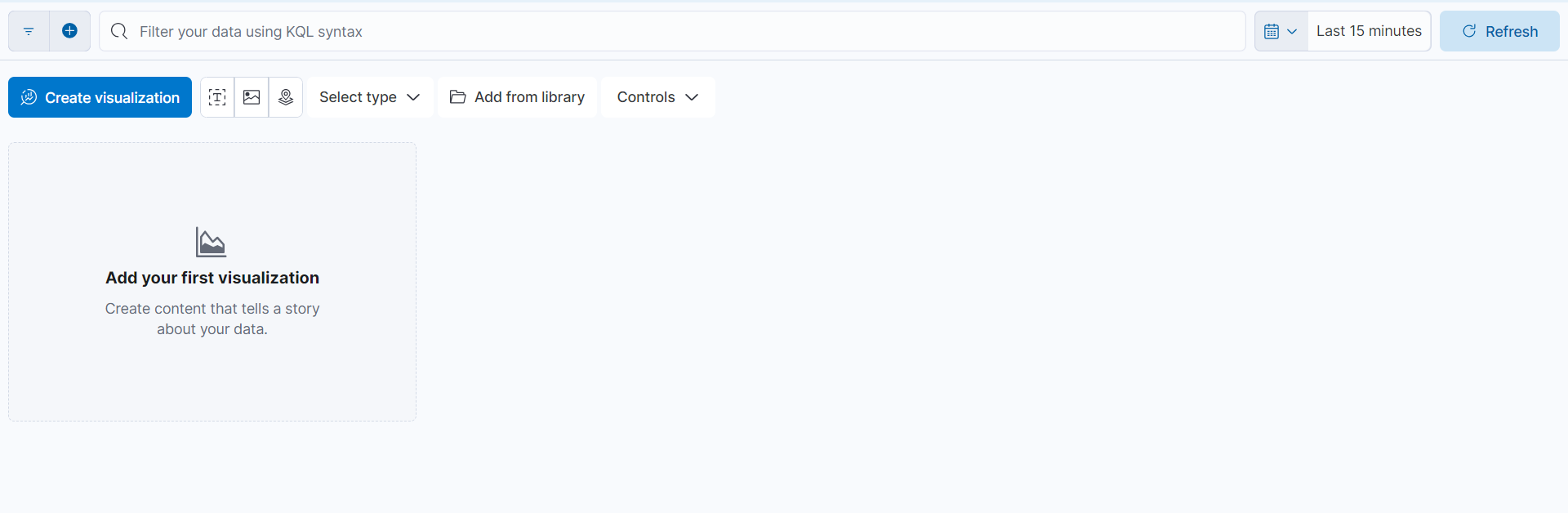


A gauche on a la liste des indice Elasticsearch disponible, pour notre cas nous allons compter sur l’indice « history\_adj\_close » ou on trouve le prix de clôture apres ajustement des actions des entreprises qu’on compare.

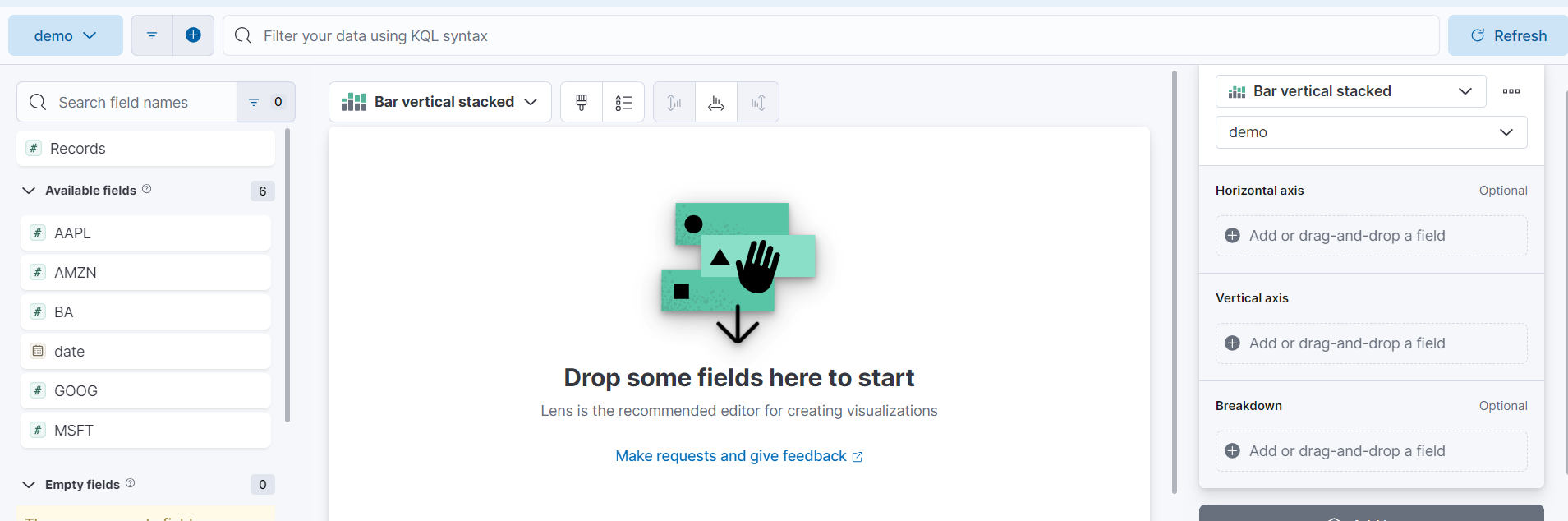
Une fois la vue de données est enregisté , on sera rediriger vers la page suivante



On peut maintenant créer notre première visualisation,

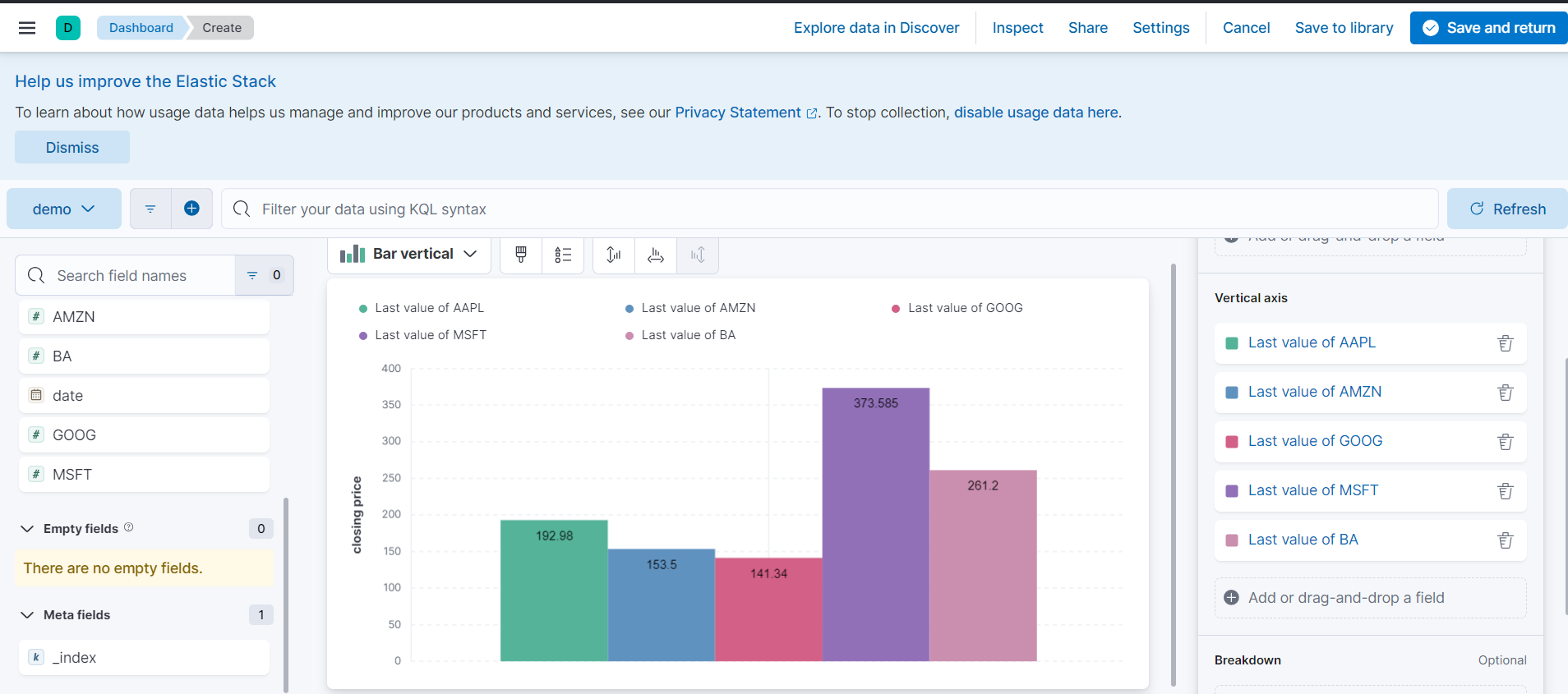


Lorsque vous allez cliquer sur créer une visualisation, vous serez redirigé vers la page suivant

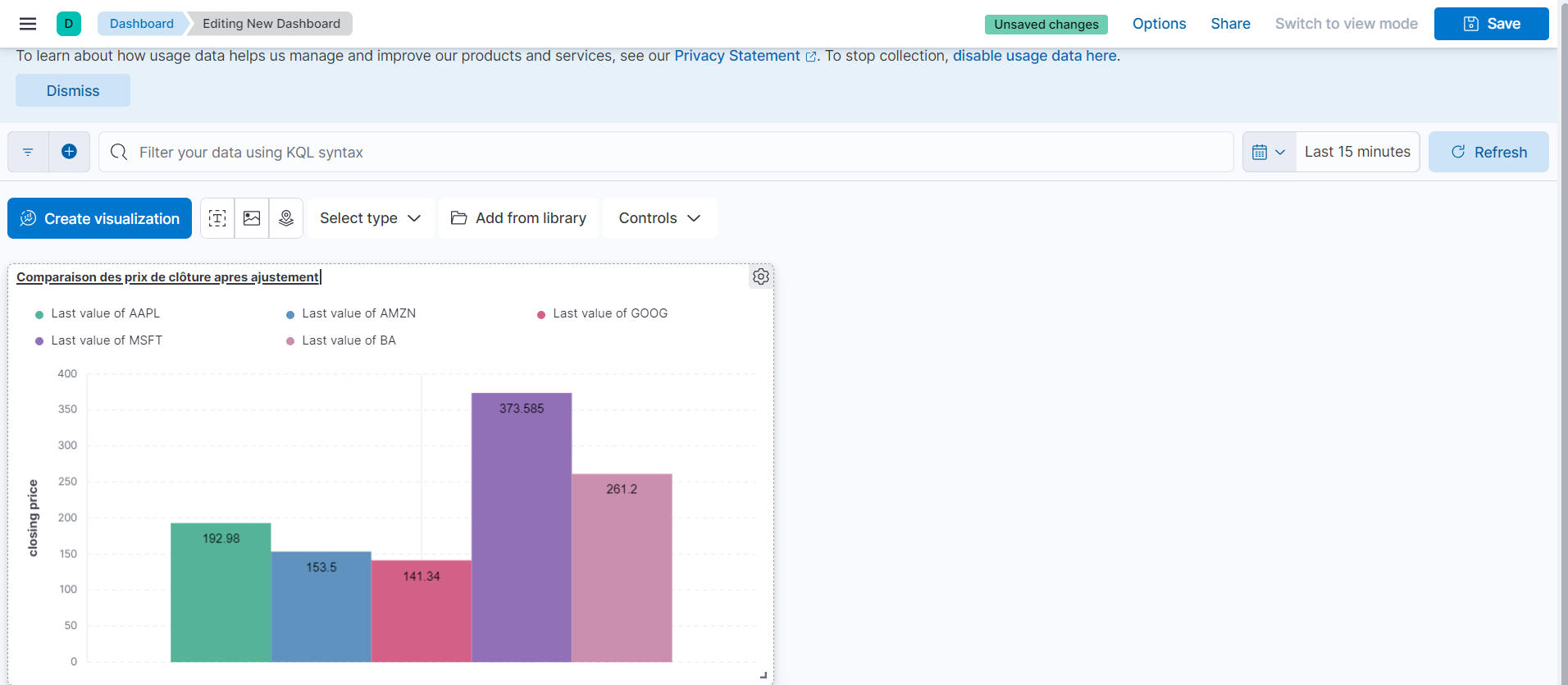


on peut glisser les champs qu’on veut visualiser parmi ceux qui existe ( à gauche )

Dans l’exemple suivant nous allons essayez de comparer la dernière valeur des prix de clôture des actions des entreprises en question



Enregistré votre visualisation et elle sera ajouté au niveau de votre Dashboard



Vous pouvez ajouter autant de visualisation que vous voulez

1. ***Conclusion***
2. ***Webographie***

***https://analyzingalpha.com/yfinance-python***

***https://docs.google.com/presentation/d/1TXQ1ybCNGyd9m0cCukneop2rBjQFqvhYgX9BFtcByEE/edit#slide=id.ge4655e227b\_0\_0***

***https://www.youtube.com/watch?v=zFFpPqHnL0o&t=489s***

***https://airflow.apache.org/docs/apache-airflow/stable/index.html***

***https://algotrading101.com/learn/yahoo-finance-api-guide/***

***https://www.youtube.com/watch?v=q8q3OFFfY6c***

***https://wire.insiderfinance.io/a-quick-and-easy-technical-analysis-project-for-beginners-using-yahoo-finance-with-python-12d042603cf9***

***https://zoo.cs.yale.edu/classes/cs458/lectures/yfinance.html***

***https://insatunisia.github.io/TP-BigData/tp4/***

***https://www.elastic.co/fr/blog/getting-started-with-the-elastic-stack-and-docker-compose***