

Département : Mathématiques et
Informatique

Challenge 1

Présenté par :

TIJJA Oumaima

Challenge 1

Proposer une solution pour mettre en œuvre un système d'analyse de sentiments basé sur une solution Big Data basée sur les commentaires des visiteurs de site web HESPRESS, qui supporte le data Ingestion, le streaming, le batch processing, et le dashboarding temps réel.

La gestion des sentiments dans les commentaires de Hespress consiste à analyser et à classer les émotions et les opinions exprimées dans les commentaires en ligne. Cela peut aider les éditeurs à comprendre l'opinion générale des utilisateurs sur un sujet donné, à mesurer la réaction des utilisateurs à un article ou à une publication, et à prendre des mesures pour améliorer l'expérience de l'utilisateur.

En général, l'analyse de sentiments se concentre sur la détection des émotions et des opinions exprimées dans du texte, telles que la colère, la joie, la tristesse, la peur, etc. Cela peut être utile pour de nombreuses applications, telles que l'analyse de la réputation en ligne, la veille médiatique, l'analyse des opinions sur les réseaux sociaux, etc.

L'analyse de sentiments est un domaine en constante évolution et les algorithmes utilisés pour détecter les sentiments évoluent constamment pour devenir plus précis et plus fiables. Cependant, il est important de noter que l'analyse de sentiments n'est pas parfaite et peut parfois donner des résultats erronés en raison de la complexité du langage humain et des contextes dans lesquels les textes sont utilisés.

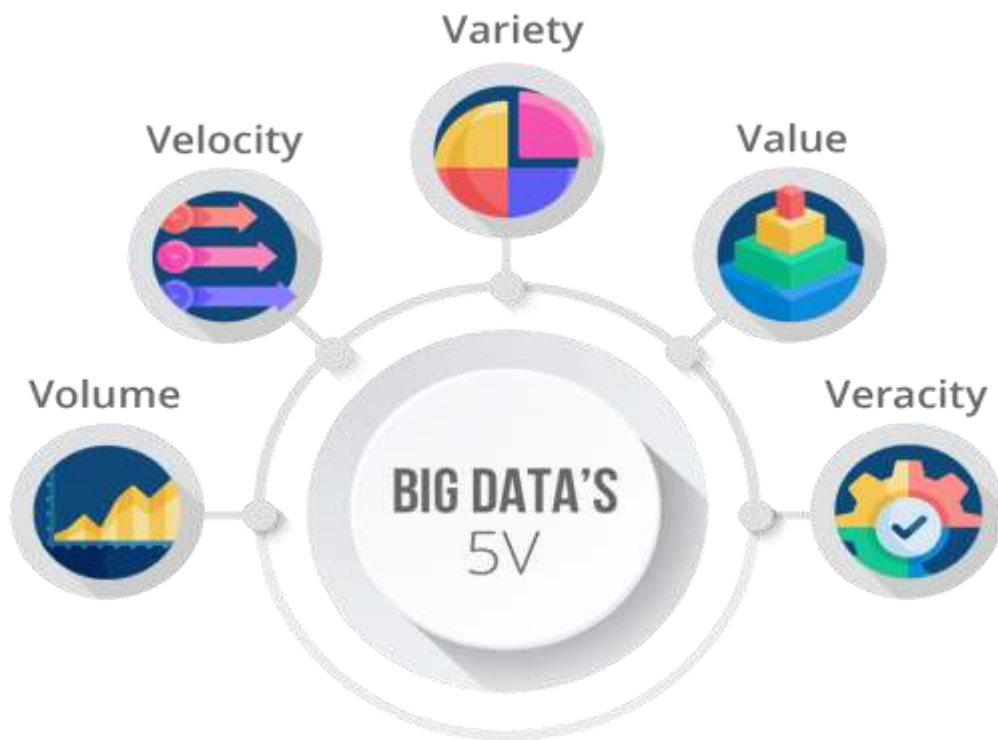
En fin de compte, l'analyse de sentiments peut être un outil précieux pour comprendre les opinions et les émotions des gens, mais il est important de l'utiliser avec prudence et de toujours vérifier les résultats avec une analyse humaine supplémentaire.



Et lorsque les commentaires sont en grande quantité, il peut être difficile de travailler avec eux manuellement. C'est là que les technologies de Big Data peuvent être utiles.

"Big data" désigne un ensemble de données volumineux et complexes qui dépasse la capacité des outils traditionnels d'analyse de données pour les gérer et les traiter efficacement. Les données peuvent provenir de sources multiples, telles que les transactions en ligne, les réseaux sociaux, les capteurs IoT, les satellites, etc.

Les 5 V du big data sont les 5 caractéristiques qui décrivent le big data :



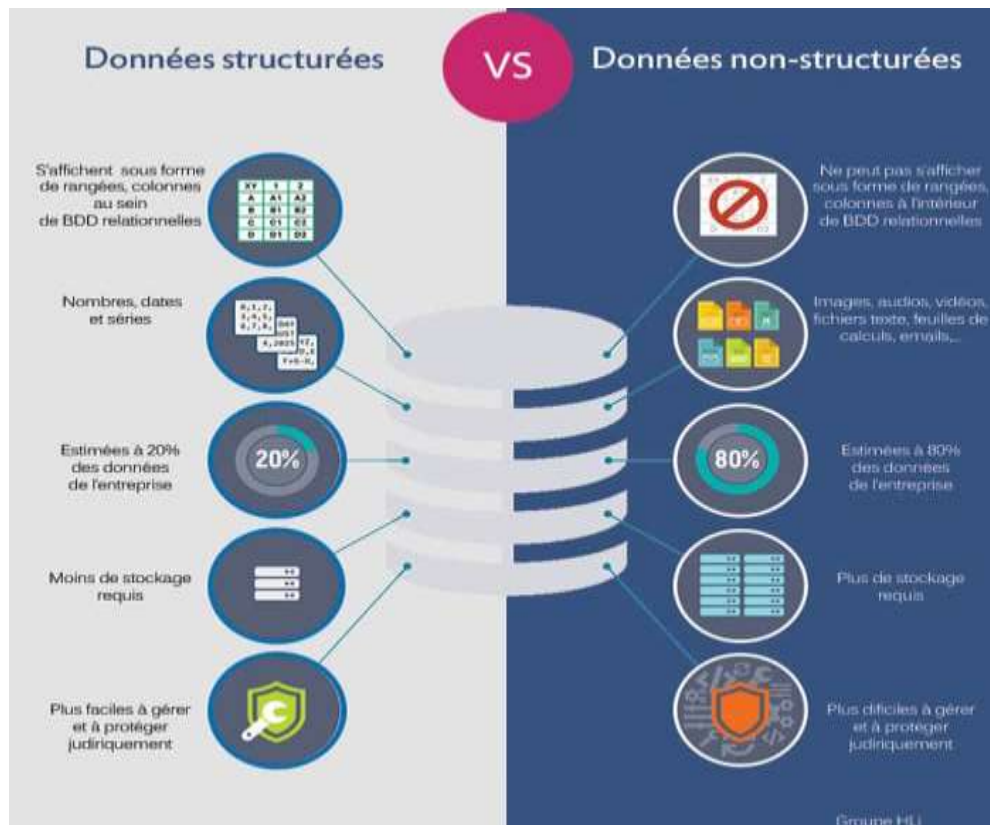
- Volume : le volume de données générées et stockées augmente exponentiellement, atteignant des tailles de plusieurs téraoctets ou pétaoctets.
- Vitesse : la vitesse à laquelle les données sont produites et collectées est très rapide, ce qui rend difficile leur traitement en temps réel.
- Variété : les données peuvent provenir de nombreuses sources et être de différents types, tels que des données structurées, semi-structurées et non structurées.

- Valeur : les données peuvent être transformées en informations précieuses pour les entreprises et les organisations qui peuvent les utiliser pour prendre des décisions informées.
- Veracité : la qualité des données peut être affectée par des erreurs, des incohérences et des données manquantes, ce qui peut rendre difficile leur analyse et leur utilisation.

L'analyse de big data peut être utilisée pour découvrir des tendances, des modèles et des corrélations cachées dans les données, ce qui peut aider les entreprises à améliorer leurs opérations, à prendre des décisions plus informées et à maximiser leur valeur.

Les sources de données peuvent être classées en trois catégories :

Structure, semi-structurées et non-structurées, en fonction de leur degré d'organisation.



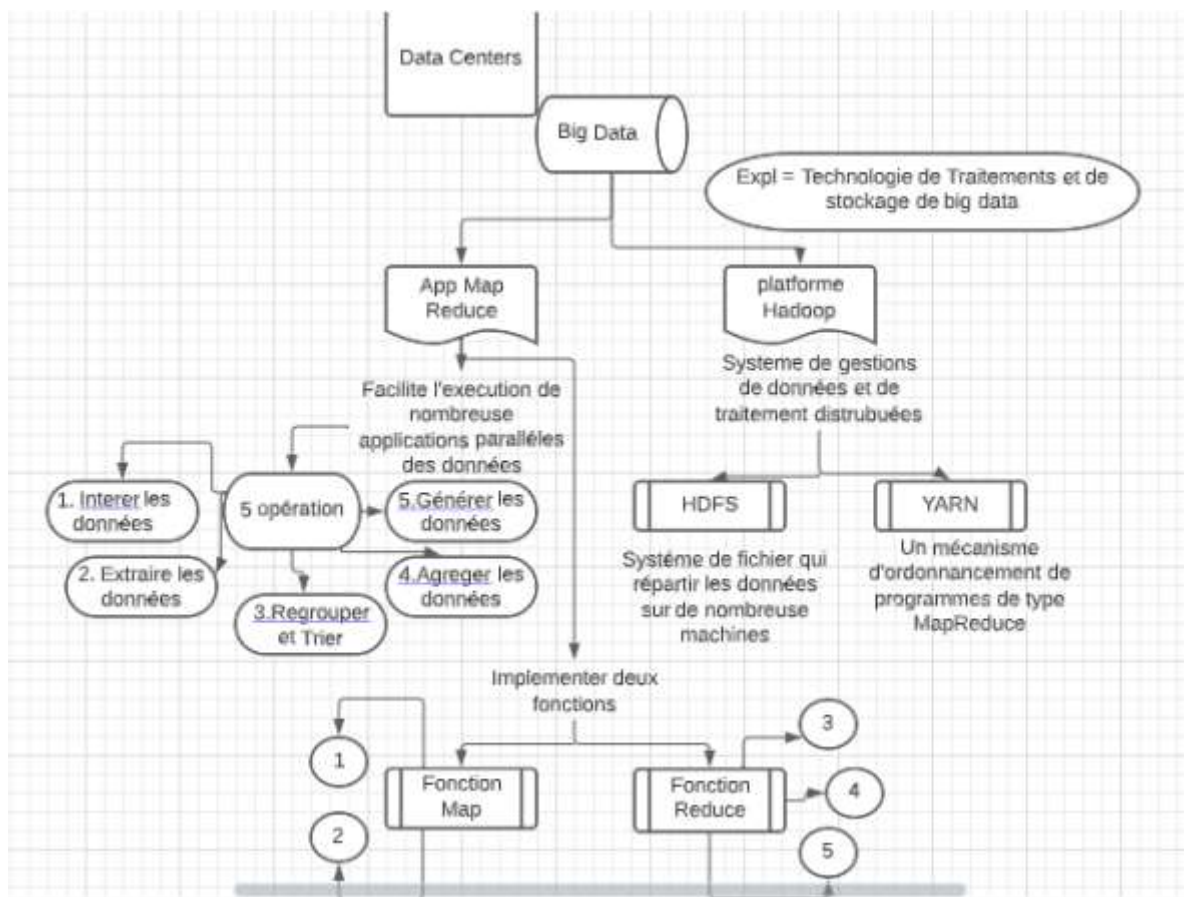
- Les données structurées sont celles qui sont organisées de manière à faciliter leur traitement et leur analyse.
- Les données semi-structurées sont celles qui ne sont pas organisées, mais peuvent être associées à des données structurées.

- Les données non-structurées sont celles qui ne sont pas organisées et n'ont pas de format défini.

Il est important de comprendre ces différences pour choisir la bonne approche pour gérer, analyser et utiliser les données de manière efficace

Alors pour proposer une solution big data en se basent sur plusieurs étapes qui va exprimer le marché, et l'étape le plus important c'est : comment va choisir l'architecture ?? car aucune technologie ne permetre de résoudre seul de problématique complexes liées à l'exploitation de données pour transformation rapides des données stockées au traitements des données et la configuration de vues complètes des données traitées.

Exploitation de données :



Outils utilises :

❖ **Apache NiFi** : Apache NiFi est un outil open source pour le traitement automatisé de données en flux. Voici les étapes générales pour travailler avec Apache NiFi :

- 1. Installation** : installez Apache NiFi sur votre système en suivant les instructions fournies dans la documentation.
- 2. Configuration de l'environnement** : configurez les sources et les destinations de données pour Apache NiFi en spécifiant les connexions à vos systèmes de données.
- 3. Création de flux de travail** : utilisez les composants Apache NiFi pour définir les étapes de votre flux de travail de données, notamment la collecte, la transformation et la distribution des données.
- 4. Exécution du flux de travail** : démarrez le flux de travail pour exécuter automatiquement les étapes définies.
- 5. Suivi des performances** : utilisez les outils de surveillance intégrés d'Apache NiFi pour surveiller les performances et les erreurs de votre flux de travail.
- 6. Maintenance et mise à jour** : maintenez et mettez à jour le flux de travail en fonction des exigences changeantes de vos systèmes de données.

❖ **Apache kafka** :

- 1.Installation de Java** : vous devez d'abord installer une installation de Java sur votre ordinateur Windows pour exécuter Apache Kafka.
- 2.Téléchargement et installation d'Apache Kafka** : vous pouvez télécharger la dernière version d'Apache Kafka sur le site Web Apache et suivre les instructions d'installation pour l'installer sur votre ordinateur Windows 8
- 3.Configurer le serveur Kafka** : une fois installé, vous devez configurer le serveur Kafka en modifiant les paramètres de configuration du fichier "server.properties".
- 4.Créer un producteur et un consommateur** : pour faire du streaming de données, vous devez créer un producteur pour envoyer les données vers le serveur Kafka et un consommateur pour recevoir les données à partir du serveur. Démarrage du serveur et envoi des données : une fois que vous avez créé un producteur et un consommateur, vous pouvez démarrer le serveur Kafka

et envoyer des données en utilisant le producteur. Le consommateur peut alors recevoir les données en streaming à partir du serveur Voici les commandes :

- --Commands to run Kafka • start Server Zookeeper : zookeeper-server-start.bat C:\kafka\config\zookeeper.properties
- start Server Kafka : kafka-server-start.bat C:\kafka\config\server.properties
- create Topic : kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 - - partitions 1 --topic comments_hespress
- - create Producer :kafka-console-producer.bat --broker-list localhost:9092 --topic comments_hespress
- - create Consumer : kafka-console-consumer.bat --bootstrap-server localhost:9092 -- topic comments_hespress

❖ Apache Spark :

Est un moteur de traitement de données en masse open source développé pour gérer les données complexes et les tâches de calcul intensives de manière plus rapide et plus efficace. Il peut être utilisé pour traiter des données en temps réel, en batch ou en streaming. Il est conçu pour être hautement évolutif et peut s'exécuter sur des clusters de nœuds pour traiter des quantités énormes de données.

Spark offre une variété de fonctionnalités pour travailler avec les données, telles que :

- **Traitement de données en temps réel** : Spark peut traiter les données en temps réel provenant de diverses sources, telles que les sockets, les fichiers, les bases de données et les messageries.
- **Traitement de données en batch** : Spark peut lire les données en batch depuis des sources telles que HDFS, HBase et Cassandra.
- **Traitement de données en streaming** : Spark peut traiter les données en streaming en temps réel à partir de sources telles que Kafka, Flume et Kinesis.
- **Machine learning** : Spark offre une bibliothèque MLlib pour effectuer des analyses de données complexes telles que la classification, la régression, le clustering, etc.

➤ **Graphiques :** Spark offre une bibliothèque GraphX pour travailler avec les données de graphe.

Spark est compatible avec de nombreux autres outils big data, tels que Hadoop, Hive et HBase, ce qui le rend facile à intégrer dans des environnements big data existants. Il peut être programmé en utilisant de nombreux langages de programmation, notamment Python, Scala, Java et R.

Choisir l'architecture :

Le choix d'une architecture pour le big data dépend de plusieurs facteurs tels que les exigences en matière de performances, la complexité des données, les coûts et les contraintes en matière de sécurité.

Les architectures de Big Data désignent l'ensemble des systèmes, des technologies et des méthodes utilisés pour collecter, stocker, traiter et analyser de grandes quantités de données, souvent appelées données volumineuses, complexes et diversifiées. Les architectures de Big Data sont conçues pour gérer les défis posés par les données Big Data, notamment la scalabilité, la disponibilité, la vitesse d'analyse et la fiabilité.

Il existe plusieurs architectures de Big Data, mais trois sont souvent considérées comme les plus importantes :

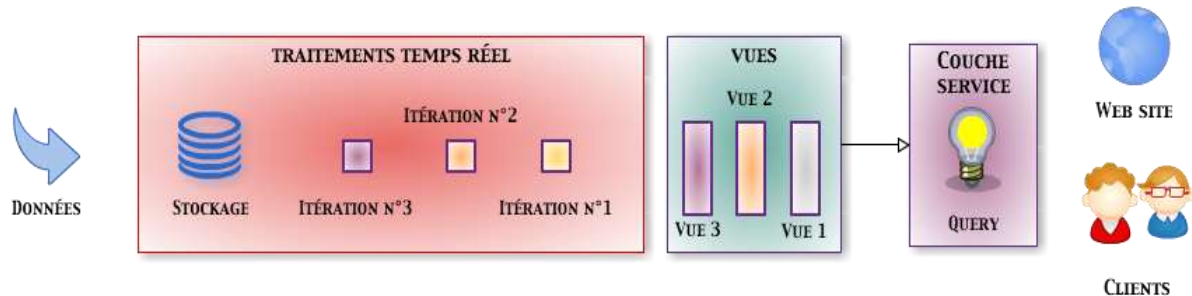
- Architecture Lambda : Cette architecture Big Data se concentre sur la capacité à traiter rapidement les données en temps réel, sans les stocker en premier lieu.
- Architecture Kappa : Cette architecture Big Data se concentre sur la capacité de traiter les données en temps réel en les stockant tout de suite.
- Architecture de l'entrepôt de données : Cette architecture Big Data est conçue pour stocker les données dans un seul endroit centralisé, afin de faciliter l'analyse et la prise de décision.

Ces architectures sont souvent combinées pour gérer les différents types de données et les différents besoins en matière d'analyse. Il est important de noter que les architectures de Big Data peuvent varier considérablement en fonction de l'industrie et de l'utilisation spécifique des données.

Et pour résoudre notre problèmes la meilleure architecture est : l'architecture KAPPA

L'architecture KAPPA :

ARCHITECTURE KAPPA



l'architecture Kappa a été formulée par Jay Kreps (LinkedIn) dans cet article. L'architecture Kappa est née en réaction à l'architecture Lambda et à sa complexité. Elle est née d'un constat simple : la plupart des solutions de traitement sont capables de traiter à la fois des batchs et des flux.

L'architecture Kappa est un modèle de système informatique distribué conçu pour fournir une plateforme fiable, évolutive et élastique pour les applications Big Data. Elle a été développée par Confluent, une entreprise spécialisée dans les technologies de la plateforme Apache Kafka.

Kappa Architecture est basée sur l'architecture en couches et se concentre sur la gestion de la donnée en temps réel. Il utilise un modèle de publication / abonnement pour la transmission de données et traite les données en temps réel avec des outils tels qu'Apache Kafka. Les données sont stockées dans un système de fichiers distribué tel qu'Apache Cassandra ou Apache Hadoop HDFS.

L'architecture Kappa permet donc de simplifier l'architecture Lambda en fusionnant les couches temps réel et batch. Elle apporte une autre évolution par rapport à l'architecture Lambda : le système de stockage des données est plus restreint et doit être un système de fichiers de type log et non modifiable (tel que Kafka).

Kafka ou un autre système permet de conserver les messages pendant un certain temps afin de pouvoir les retraiter. De fait, et encore plus que l'architecture Lambda, l'architecture Kappa ne permet pas le stockage permanent des données. Elle est plus dédiée à leur traitement.

- Traitement des données en temps réel
- Agrégation de journaux
- Collecte et suivi des métriques
- Recherche d'événements

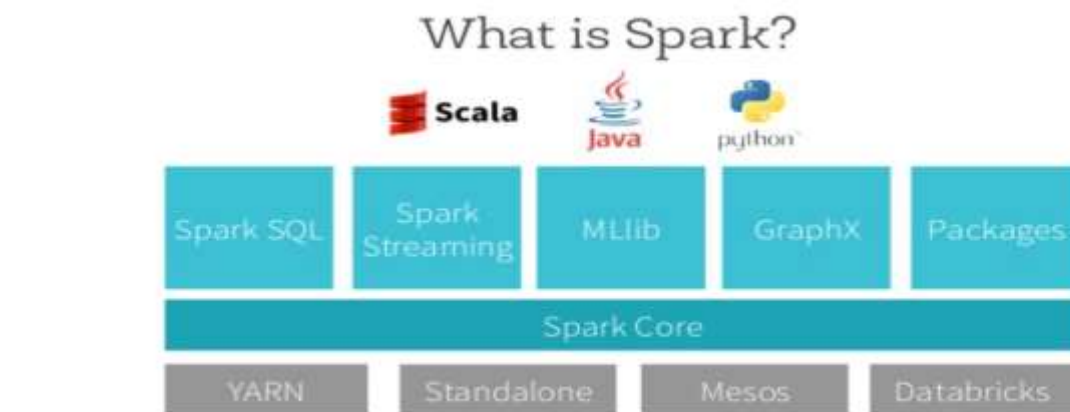
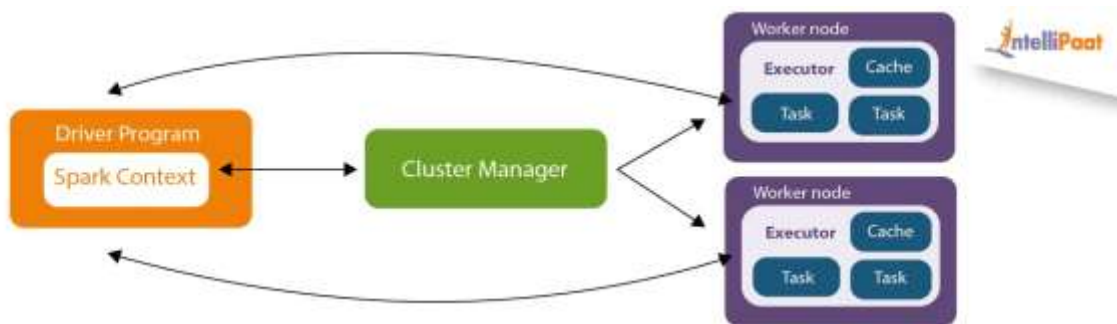
- Suivi d'activité
- Traitement de flux

Kafka dispose d'un solide écosystème d'outils et de bibliothèques qui facilitent l'intégration avec d'autres technologies et applications. Il peut être utilisé avec une variété de langages de programmation, notamment Java, Scala, Python, etc.

En résumé, Apache Kafka est un outil puissant pour gérer les flux de données en temps réel, offrant des performances, une évolutivité et une fiabilité élevées pour une variété de cas d'utilisation.

Vues : Implémenter à l'aide Apache Spark.

Apache Spark peut être utilisé pour le traitement en lots des commentaires des visiteurs en utilisant des algorithmes d'analyse de sentiments tels que le NLP pour classer les commentaires en fonction de leur polarité (positif, négatif, neutre).



Apache Spark Core : comme son nom l'indique, est le moteur d'exécution du framework, la base du framework. Il fournit la répartition des tâches distribuées, le scheduling et des fonctionnalités lectures/écritures de base. L'API RDD est implémentée (Resilient Distributed Datasets) sur Spark Core, qui est une collection logique de données partitionnées sur le cluster. Les RDD peuvent être créés de deux façons : l'une est en référençant des jeux de données dans des systèmes de stockage externes (ou bien en le créant via le SparkContext) et la seconde consiste à appliquer des transformations (Map, filtre, Reduce, jointure) sur des RDD existants.

Spark SQL : est le composant qui vient au-dessus de la couche Core et qui introduit une nouvelle abstraction de données SchemaRDD, des données structurées et semi-structurées. Spark SQL fournit des fonctions qui se basent sur Spark Core et ses APIs Scala, Java, Python, SQL et R. Spark SQL apporte une vision semi-structurée, les DataFrames, et à partir de la version 2 les Datasets, ce qui permet d'extraire, transformer et de charger des données sous différents formats (csv, Json, Parquet, base de données). La brique Spark SQL s'intègre très bien avec Apache Hive (en utilisant le langage HQL) et aussi avec des connexions JDBC pour se connecter à des serveurs de BI.

Spark Streaming : est la partie traitement en pseudo temps réel d'Apache Spark. La brique Streaming est basée sur Spark Core et traite la donnée sous forme de mini-batches espacés par un instant T. L'API de Spark Streaming construit des RDD de type spécifique (des Dstreams) mais les traitements qui seront faits sur les Dstreams sont identiques à ceux de Spark Core classique. Spark Streaming supporte plusieurs sources de données, tel qu'Apache Kafka, Flume, Kinesis...

MLlib (bibliothèque Machine Learning) : est une bibliothèque de Machine Learning distribuée et ses algorithmes sont conçus pour être exécutés sur un cluster de machines d'une manière distribuée. MLlib utilise des RDD aux types spécifiques. Les inputs des algorithmes doivent être des RDD[Vector] (pour des données n'ayant pas de label), des RDD[LabeledPoint] (spécifique à l'apprentissage supervisé) ou bien des RDD[Rating] (pour les systèmes de recommandation). Il faut savoir qu'il existe aussi une autre bibliothèque de Machine Learning sous Spark, Spark ML, qui elle est basée sur l'API Dataframe.

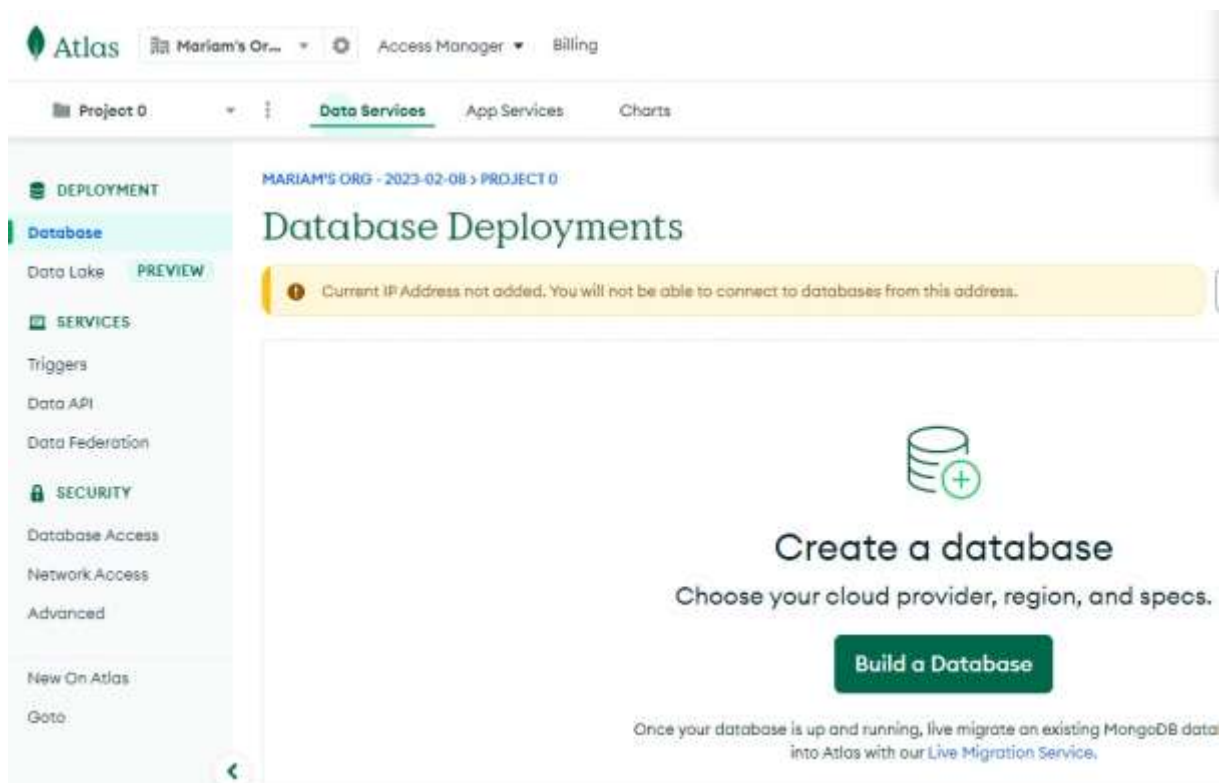
GraphX : est un framework de traitement de graphe distribué sur Spark. Il fournit une API pour faire du calcul graphe qui peut modéliser les graphes définis par l'utilisateur en utilisant l'API d'abstraction Pregel. Il fournit également une durée d'exécution optimisée pour cette abstraction.

Résultat Final :

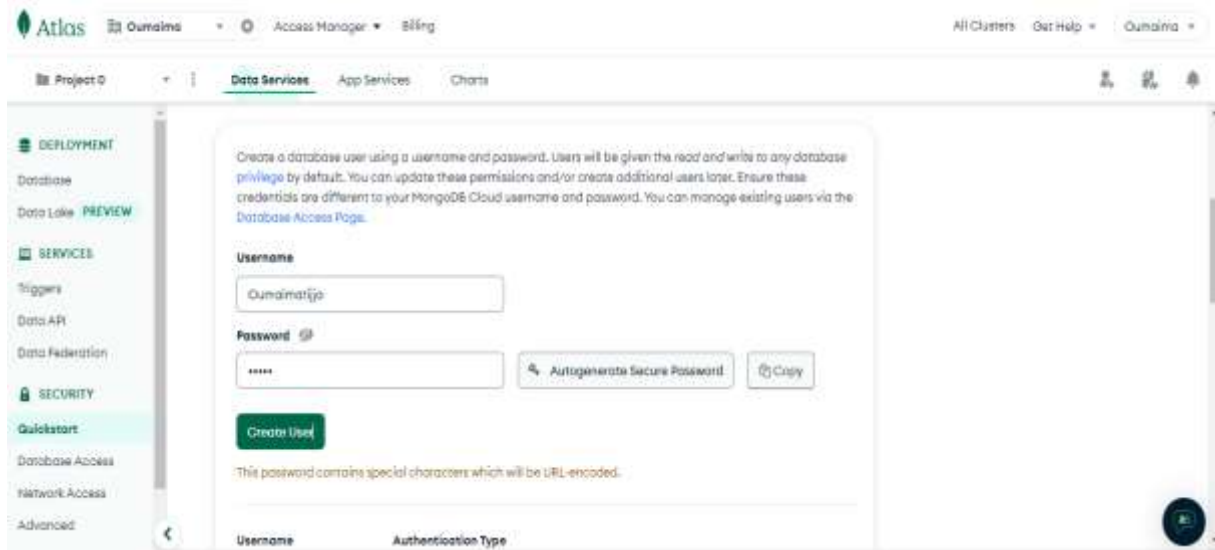
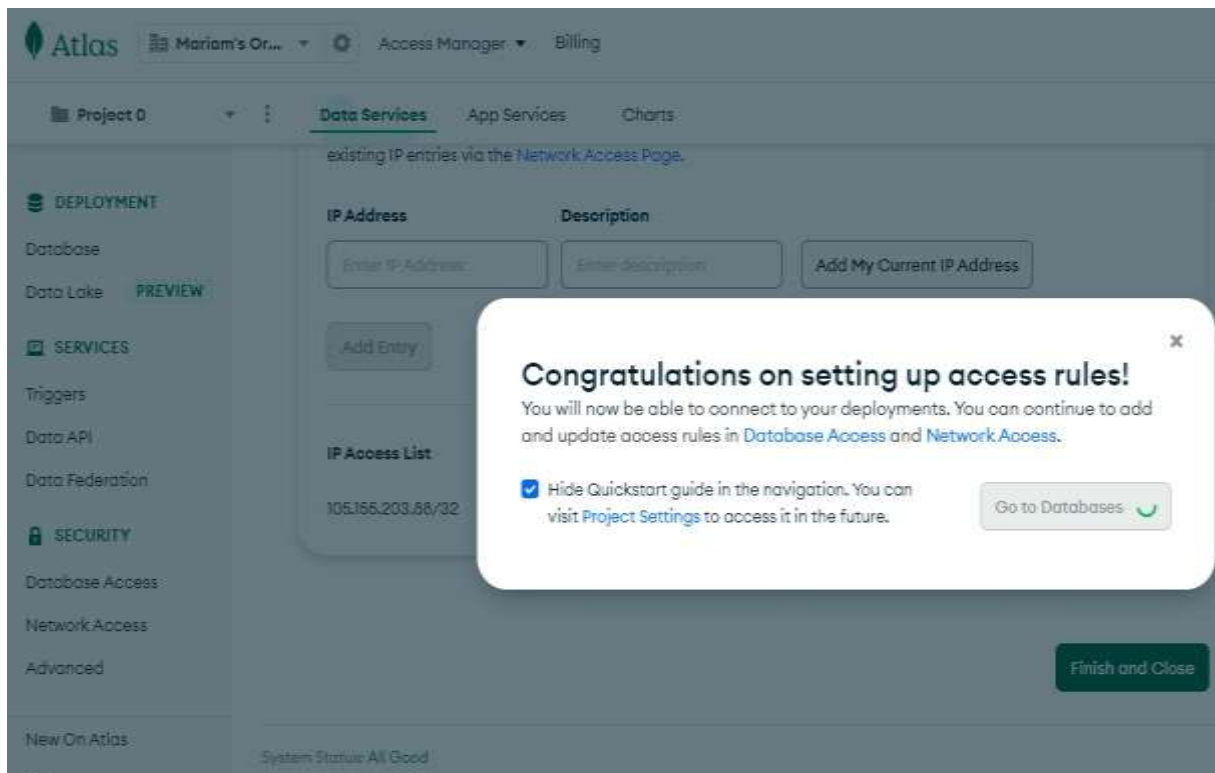
- Exécuter des tâches dans les nœuds et les transformations.
- Communiquer avec le programme pilote.
- Gérer la mémoire.

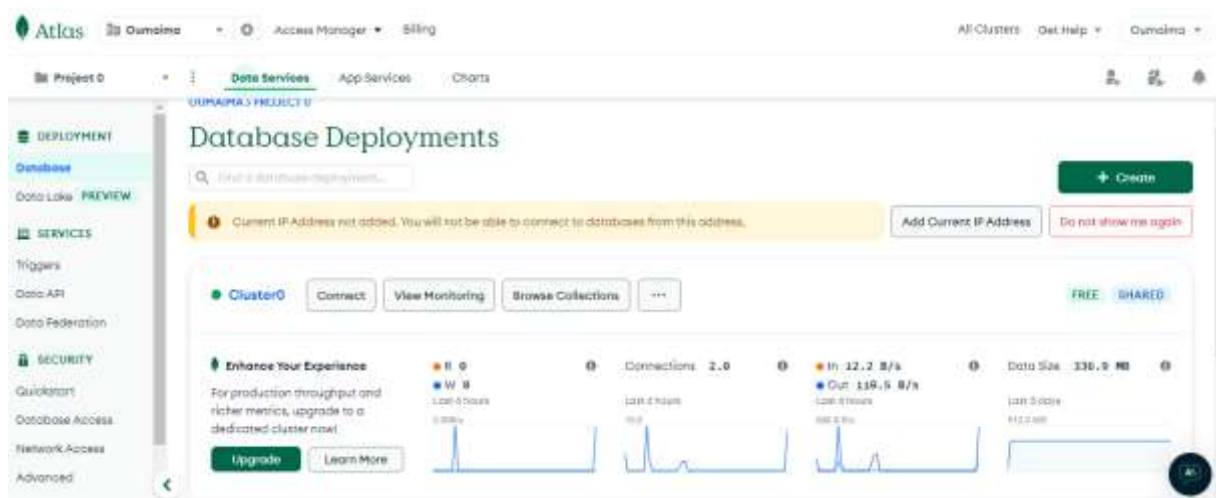
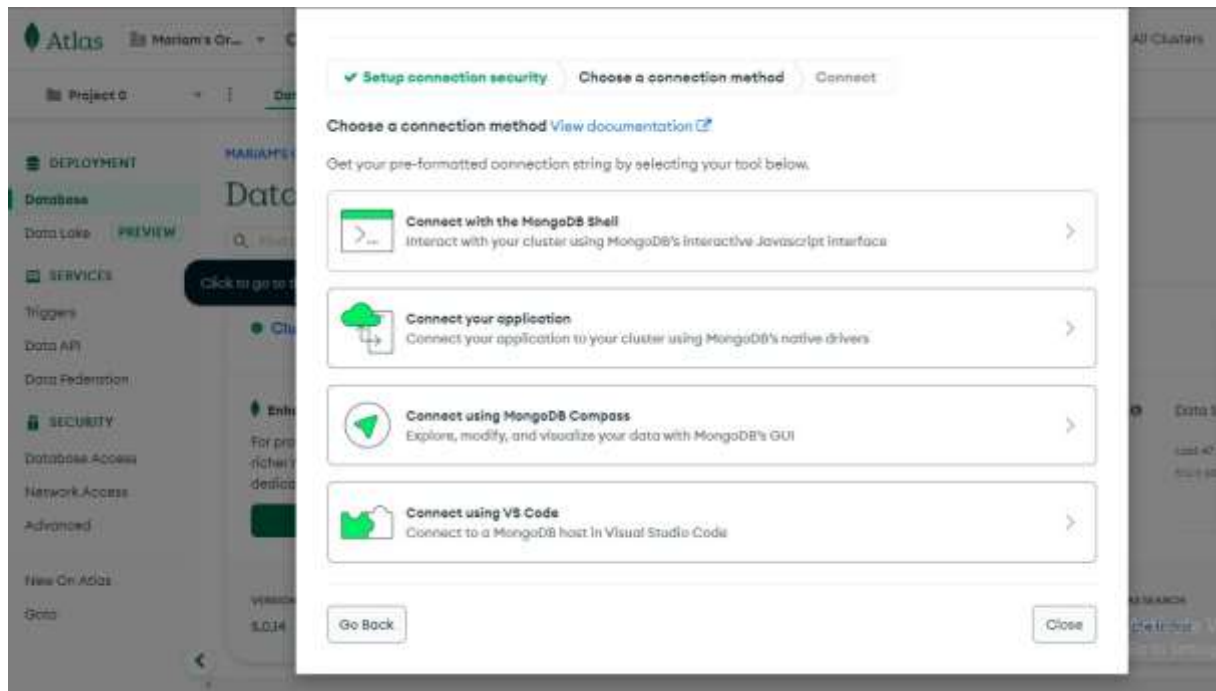
Extraction des commentaires

- Création de compte MongoDB :



- Entrer userName et password :



Et Après exécution de code de web scraping on obtient les commentaires :

Le web scraping est un processus consistant à extraire des données à partir d'une source en ligne, généralement un site web. Il implique la collecte automatisée de données à partir d'un site web en utilisant des scripts ou des programmes spécifiques, tels que des robots d'exploration, des scrappers ou des extracteurs de données. Les données extraites peuvent inclure des informations telles que du texte, des images, des tableaux, des fichiers audio et vidéo, etc. Les données peuvent ensuite être utilisées pour des analyses, des visualisations, des études de marché, des algorithmes de recommandation, etc.



Data ingestion

Ingestion de données : Les commentaires des visiteurs peuvent être collectés en temps réel via Apache NiFi pour l'ingestion de données en flux. ➤ Pour importer les données de MongoDB vers Apache Kafka en utilisant Apache NiFi, vous pouvez suivre les étapes suivantes :

- 1. Configuration de MongoDB :** Assurez-vous que MongoDB est configuré et en cours d'exécution. Vérifiez également que vous avez accès aux informations de connexion à la base de données, telles que l'adresse IP, le nom d'utilisateur et le mot de passe.
- 2. Installation et configuration d'Apache NiFi :** Installez Apache NiFi sur votre système et configurez-le en fonction de vos besoins.
- 3. Ajout de la source MongoDB :** Dans Apache NiFi, ajoutez un nouveau composant "GetMongo" à votre flux de travail pour accéder aux données de MongoDB. Configurez les informations de connexion à la base de données en utilisant les informations de connexion obtenues à l'étape 1.
- 4. Ajout de la destination Kafka :** Ajoutez un nouveau composant "PutKafka" à votre flux de travail pour définir la destination des données vers Apache Kafka. Configurez les informations de connexion à Apache Kafka, telles que l'adresse du serveur Kafka et le nom du sujet.
- 5. Configurez le format des données :** Configurez le format des données que vous souhaitez envoyer à Apache Kafka. Par exemple, vous pouvez utiliser le format JSON ou Avro pour les données.

6. Exécution du flux de travail : Lorsque vous avez terminé de configurer votre flux de travail, vous pouvez le démarrer pour importer les données de MongoDB vers Apache Kafka.

Le streaming

Streaming : Apache Kafka peut être utilisé pour gérer les données en temps réel en les publiant sur un canal et en les souscrivant à partir de là pour le traitement en temps réel.

Pour travailler avec Apache kafka :

1. Installation de Apache Kafka : Téléchargez et installez Apache Kafka sur votre système. Vous pouvez installer Apache Kafka en utilisant les packages de distribution pour votre système d'exploitation ou en téléchargeant le code source et en le compilant.

2. Configuration du cluster Kafka : Configurez les nœuds du cluster Kafka pour définir les paramètres tels que les ports, les répertoires de données, etc.

3. Création de sujets : Créez des sujets pour organiser vos données. Les sujets sont utilisés pour publier et consommer les données dans Apache Kafka.

4. Écriture de données : Écrivez des données dans les sujets en utilisant un producteur Kafka. Les données peuvent être envoyées à Apache Kafka en utilisant un format tel que JSON, Avro, Protobuf, etc.

5. Lecture de données : Consommez les données à partir des sujets en utilisant un consommateur Kafka. Les données peuvent être lues en temps réel ou en mode batch.

```
C:\kafka\bin\windows>kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic comment_hespress
WARNING: Due to limitations in metric names, topics with a period (".") or underscore ("_") could collide. To avoid issues
it is best to use either, but not both.
Created topic comment_hespress.
```

- **Create producer :**

```
C:\kafka\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic comments_hespress  
>
```

- **Create cosumer :**

```
C:\kafka\bin\windows>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic comments_hespress
```