COMPTE RENDU DU PROJET DE 5DATA

Membres du groupe :

Emmanuel N'GUESSAN

Elvis AKOTEGNON

Marvel NGANKAM

Ibrahim BAH-SALIFOU

OBJECTIF DU PROJET

Il nous a été demandé au cours de ce projet de mettre sur pieds une plateforme big-data permettant d'effectuer des traitements notamment collecte, nettoyage, stockage et analyse d'un ensemble de données issus des étudiants de Supinfo. Ces données devraient servir à prévoir les comportements des étudiants ainsi que de créer des relations qui vont être utilisés par l'école en fonction de certains critères à savoir leurs régions d'origines, leurs établissements d'origines, leur mobilité, l'entreprise dans laquelle ils sont en contrat pro, leur année diplômante, les différents stages qu'ils ont eu à effectuer et les entreprises dans lesquelles ils les ont fait, et enfin les études qu'ils ont eu à effectuer au cours de leur cursus.

ARCHITECTURE

Notre architecture est constituée de trois services principaux à savoir d'un cluster Spark, d'un Cluster mongo et d'un Cluster Chart. Pour la mise en place des différents clusters la technologie utilisée est Docker. Le choix de Docker ici s'est avéré très intéressant dans la mesure où il nous permet de créer des environnements de travail adapté pour le travail que nous voulons faire. Contrairement aux machines virtuelles que nous aurions également pu utiliser les containers que nous utilisons avec docker sont bien plus légers et rapide ce qui rend le déploiement de notre application facile. A l'aide des Dockerfile on crée des images qui serviront à mettre en place des containers contenant l'architecture des différents clusters.

CHOIX DE LA BASE DE DONNEES NOSQL

La base de données NoSQL que nous avons choisi ici est MongoDB d'une part parce que le format utilisé par MongoDB pour stocker les données est le JSON qui est un format très facile à utiliser. De plus étant donné qu'il nous a été demandé de mettre sur pieds une application Big-data, et qui dit « Big-data » dit important volume de données, MongoDB est très utilisé pour la mise en place des applications big-data en temps-réel. C'est également celui que nous maîtrisons le mieux.

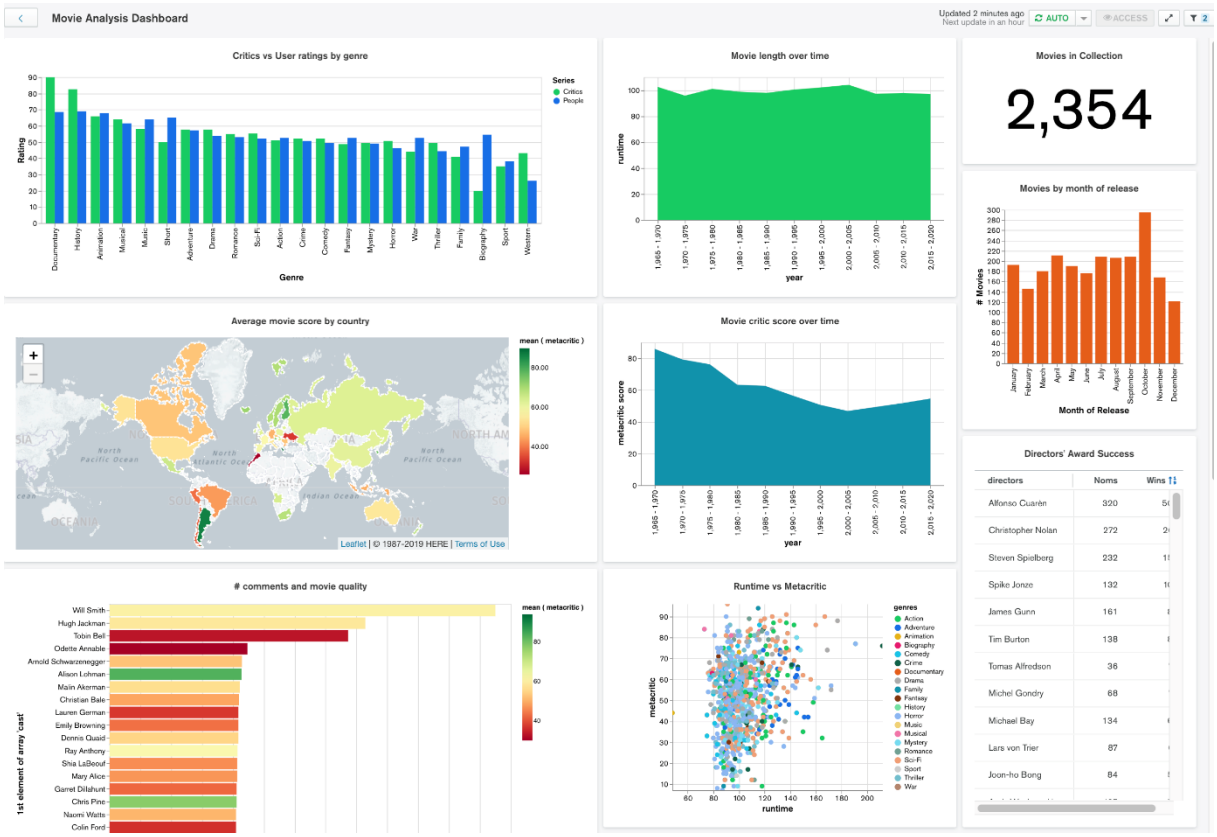
TECHNOLOGIES UTILISEES

- **Python** : Python est un langage de programmation interprété, multiparadigme et multiplateformes permettant de travailler de manière efficace et rapide. Il contient de nombreuses librairies adaptées pour les opérations de machine Learning et de data science. Nous avons utilisé python ici pour faire des scripts notamment celui de la génération des données, ainsi qu'effectuer le traitement de données sous **Apache Spark**.
- **Docker** : **Docker** est une technologie de conteneurisation qui permet la création et l'utilisation de conteneurs Linux. Nous avons utilisé Docker pour mettre en place notre architecture car comme nous l'avons dit plus haut en termes de flexibilité et de ressources les containers présentent un gros avantage. Il nous a également permis de créer des réplicas de données au niveau de notre base de données MongoDB.

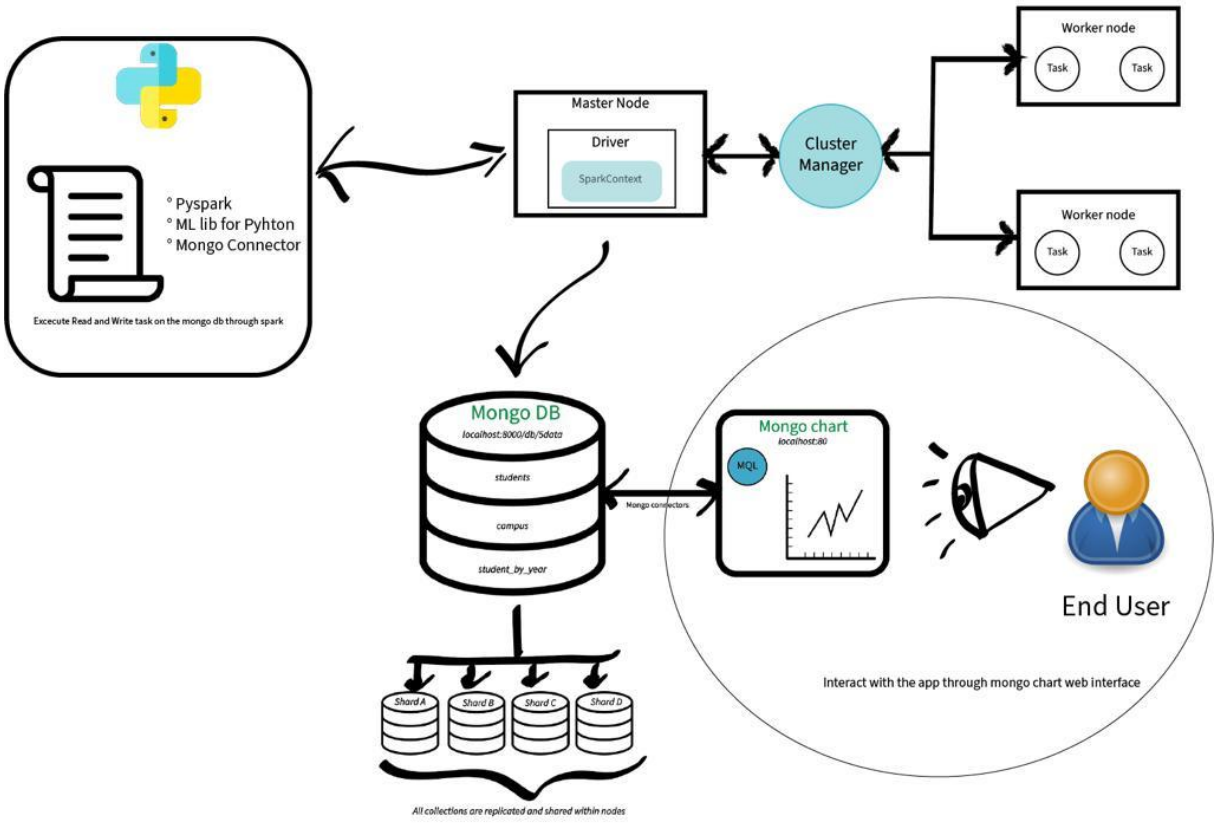
- **Apache Spark** : c'est un outil permettant d'effectuer des opérations de data engineering, de data science, et de machine Learning sur des

machines ou des clusters à nœuds uniques. Il permet également d'effectuer des requêtes SQL de manière rapide et distribué pour du dashboarding et du ad hoc reporting. Ici On a utilisé Spark pour gérer le flow de données et effectuer le traitement des données en temps réel.

- **MongoDB Charts** : **MongoDB Charts** est un outil permettant de créer des représentations visuelles des données issus de notre base de données **MongoDB**. Nous avons utilisé MongoDB Charts pour une effectuer une représentation visuelle de nos données et mettre l'accent sur les différentes corrélations entre les variables au niveau de nos données et ainsi observer les tendances. Ci-dessous quelques d'illustrations de données que nous pouvons avoir avec MongoDB Charts.



- **MongoDB** : C'est un système de gestion de base de données NoSQL très utilisé, qui stocke les données sous forme de documents BSON (binary JSON). C'est un magasin de données CP c'est-à-dire qu'il résout les partitionnements de réseau en maintenant la cohérence au détriment de la disponibilité. Très fréquemment utilisé pour les applications de big-data. Il utilise un système à nœud unique.


Worker node
Task
Task
Master Node
Driver
SparkContext
Cluster Manager
Worker node
Task
Task
° Pyspark
° ML lib for Pyhton
° Mongo Connector
Excecute Read and Write task on the mongo db through spark
Mongo DB
localhost:9000/db/5data
students
campus
student_by_year
Mongo connectors
Mongo chart
localhost:80
MQL
End User
Interact with the app through mongo chart web interface
Shard A
Shard B
Shard C
Shard D
All collections are replicated and shared within nodes

Voici ci-dessus le schéma illustrant notre architecture. Tout à gauche nous avons le langage de programmation python qui a été utilisé pour les opérations lecture et d'écriture sur notre base de données MongoDB à travers Spark. Pyspark qui est une interface Spark en python a été utilisé ici pour nous connecter à Spark afin de lancer les traitements sur les données. Ensuite nous comme nous pouvons le voir sur le schéma nous avons un nœud master relié à Spark et à MongoDB permettant aux deux de communiquer ensemble. Ainsi les données traitées grâce à SPARK seront directement acheminées en base de données grâce au Master Node pour

être stockées. Au niveau du Master Node nous avons crée deux Worker Node qui sont gérés par un cluster Manager permettant d'assurer la haute disponibilité. Plusieurs réplicas ont été crée au niveau de la base de données MongoDB grâce à Docker. Enfin les différentes données stockées en base de données vont être affichées sous forme de graphe grâce à Mongo Chart.