Documentation Technique

Crytpo VIZ

Objet du document : Documentation Technique

Nom du document : DT Crypto VIZ V1A

Version	Rédacteur	Date
1A	Amine	17/02/2024

Table des matières

1. INTRODUCTION

Le voyage de l'équipe CryptoViz à travers l'univers de la technologie de streaming de données a été une aventure passionnante, marquée par une exploration approfondie d'Apache Kafka et une adaptation stratégique à nos découvertes et défis techniques. Ce document détaille notre parcours, nos choix méthodologiques, et nos décisions architecturales, en mettant un accent particulier sur notre gestion des topics Kafka et le traitement des données des crypto-monnaies.

2. ORIGINES ET EVOLUTION DU PROJET

2.1 Immersion dans Apache Kafka

Le projet a démarré sur une note d'exploration, avec chaque membre plongeant dans les méandres d'Apache Kafka pour en démêler les principes fondamentaux. Cette phase d'apprentissage a été cruciale, nous armant de la connaissance nécessaire pour utiliser Kafka comme colonne vertébrale de notre système de traitement de données en temps réel.

2.2 Première Approche avec Google Cloud Plateforme (GCP)

Nous avons opté pour une machine virtuelle sur Google Cloud Platform (GCP) pour déployer notre infrastructure. Initialement, nous avons envisagé d'intégrer Apache Spark pour la transformation des données après leur collecte. Cette décision nous a conduit à configurer une infrastructure complète basée sur Hadoop YARN et HDFS, envisageant Spark comme moteur de traitement. Cependant, nous avons perdu l'accès à cette VM, ce qui nous a obligés à revoir notre stratégie.

2.3 Ajustement et Simplification sur GCP

Cela étant, nous avons pris la décision stratégique de simplifier notre approche. Nous avons recentré nos efforts sur Kafka, éliminant la complexité ajoutée par l'intégration de Spark, pour nous concentrer sur la collecte et le traitement direct des données via Kafka.

2.4 Adoption du Format JSON pour le Stockage des Données

Contrairement à notre plan initial d'utiliser des fichiers CSV, nous avons opté pour le format JSON pour l'enregistrement des données. Ce choix a été motivé par la flexibilité du format JSON, facilitant l'intégration et l'indexation des données pour des analyses futures, tout en offrant une structure claire pour le stockage des informations complexes des cryptomonnaies.

2.5 Architecture Profonde des Topics Kafka

Conception et Rationalisation des Topics
Notre architecture Kafka repose sur deux topics principaux : datacrypto pour les flux de données des cryptomonnaies et blogtopic pour les nouvelles et analyses du marché. Ce design vise à segmenter clairement les types de données traitées, optimisant ainsi la gestion et le traitement spécifique à chaque catégorie d'informations.

2.6 Topic datacrypto : Une Analyse Approfondie

Le topic data crypto a été méticuleusement conçu pour capturer les nuances du marché des crypto-monnaies. Chaque message dans ce topic contient des données précieuses sur les crypto monnaies, notamment :

Nom de la Crypto : Identifie la cryptomonnaie,
Prix : Le prix actuel de la crypto monnaie en USD,
Market Cap (Capitalisation Boursière) : La valeur totale du marché de la cryptomonnaie,
Temps : Timestamp ajouté par le producteur pour marquer le moment de la collecte des données,
ID : Un identifiant unique ajouté par le consommateur pour chaque entrée, facilitant l'indexation et la recherche dans le fichier JSON.
Cette structure de données enrichie permet une analyse détaillée et une visualisation précise des tendances du marché des crypto-monnaies.

2.7 Choix Technique et Optimisation des Ressources

En configurant les topics avec un facteur de réplication de 1 et 2 partitions, nous avons visé un équilibre entre robustesse, performance et économie des ressources. Cette configuration assure une haute disponibilité des données et une distribution équilibrée des charges, tout en étant consciente des limitations de notre environnement de déploiement dont.

3. NOTES

- Choix de Kafka pour la gestion des flux de données : Nous avons opté pour Apache Kafka en raison de sa haute performance, sa fiabilité et sa capacité à gérer des volumes massifs de données en temps réel. Cette décision technique s'appuie sur l'architecture distribuée de Kafka, qui permet une scalabilité horizontale, essentielle pour notre objectif de capturer et de traiter des flux continus de données cryptographiques.

- Implémentation d'une architecture à microservices: En utilisant Kafka comme colonne vertébrale de notre infrastructure, nous avons pu adopter une architecture à microservices pour notre application. Cette approche nous permet de découpler les services de collecte de données, de traitement et de stockage, facilitant ainsi la maintenance, l'évolutivité et la mise à jour des composants individuels sans perturber le système global.

- Stratégie de partitionnement et de réplication: Nous avons défini les topics Kafka avec une stratégie de partitionnement et de réplication soigneusement considérée, choisissant deux partitions et un facteur de réplication de deux pour équilibrer la charge et assurer la haute disponibilité. Cette configuration optimise le parallélisme dans le traitement des données et augmente la résilience du système face aux pannes de nœuds.

- Utilisation de la sérialisation JSON pour les messages Kafka: Le choix de JSON pour la sérialisation des messages nous permet de maintenir une structure de données flexible et aisément interprétable, ce qui est crucial pour l'intégration avec divers consommateurs et pour faciliter le traitement des données, notamment pour l'analyse et la visualisation.

- Gestion fine des délais entre les appels API: Afin de respecter les limitations de l'API CoinGecko et de prévenir la surcharge, nous avons délibérément configuré notre producteur Kafka pour faire des requêtes toutes les 20 secondes. Cette précaution technique assure un flux de données stable et durable sans compromettre la disponibilité de l'API source.

- Décision contre l'usage d'une base de données traditionnelle : Après évaluation, nous avons conclu qu'une base de données traditionnelle n'était pas nécessaire pour nos besoins. Le stockage des données transformées directement dans des fichiers JSON plutôt que dans une base de données PostgreSQL ou similaire nous permet de simplifier notre architecture et de réduire les coûts et la complexité de gestion, tout en répondant efficacement à nos besoins d'analyse et de visualisation des données.

- Conception soignée des topics Kafka : En créant deux topics distincts, l'un pour les données cryptographiques et l'autre pour les nouvelles du blog, nous avons pu segmenter et organiser efficacement les flux de données. Cette segmentation facilite le traitement ciblé des données et améliore la clarté de l'architecture du système, permettant une gestion plus fine des consommateurs Kafka selon leur domaine d'intérêt spécifique.