

Projet de semestre

- ☐ [Ajouter la structure de la documentation](#)
- ☐ [Description des routes de l'API](#)

Introduction

Le trading de cryptomonnaies se démocratise de plus en plus notamment grâce à la multiplication des plateformes de trading comme Binance, Coinbase ou encore Crypto.com. Ces plateformes permettent d'acheter et de vendre des cryptomonnaies assez facilement notamment grâce à leurs applications sur smartphone. Cette accessibilité et cette simplification de l'achat et revente de cryptomonnaies permet à un plus grand public, même débutant, d'utiliser et de posséder des cryptomonnaies.

Avec une augmentation de 33% sur un an, le nombre de personnes détenant des cryptomonnaies dans le monde atteint 562 millions en 2024 contre 420 millions en 2023.

(<https://journalducoin.com/actualites/bilan-adoption-cryptos-monde-2024-rapport-triple-a/>) Cette augmentation de l'utilisation des cryptomonnaies montre que l'intérêt pour les cryptomonnaies augmentent que ce soit pour un simple investissement en vue de faire un profit ou même, pour certains, d'avoir une alternative aux monnaies fiduciaires.

Beaucoup de personnes remarquent la grande volatilité des prix des cryptomonnaies et font du profit grâce à ces variations de prix. Cependant, bien que certains fassent effectivement des bénéfices, d'autres font de grosses pertes. Et c'est pourquoi les simulateurs de trading de cryptomonnaies sont utiles. Ces simulateurs permettent de s'entraîner avec les stratégies existantes et de mieux comprendre le marché des cryptomonnaies sans pour autant avoir à réellement investir dans une cryptomonnaie et risquer de perdre leur capital investi. Ces simulateurs reproduisent les fluctuations des différentes cryptomonnaies soit en temps réel soit sur les données passées. S'entraîner sur une période passée peut paraître à première vue pas vraiment utile, mais en réalité cela peut permettre de s'entraîner sur une situation passée qui pourrait se reproduire à l'avenir. Les simulateurs sont donc aussi utiles à des débutants, pour leur faire découvrir le trading de cryptomonnaies sans leur faire perdre d'argent, ils peuvent y découvrir ce qu'est un ordre d'achat ou de vente et comment en placer, qu'à des traders plus expérimentés en leur permettant de revenir sur des périodes passées, ils peuvent y tester de nouvelles stratégies en utilisant les différents indicateurs à disposition. En réalisant ces tests sur un simulateur, il pourra par la suite voir les résultats obtenus et les appliquer dans le vrai trading de cryptomonnaies. Les simulateurs de trading de cryptomonnaies sont donc très utiles pour découvrir le trading de cryptomonnaies, pour tester et améliorer les différents outils et stratégies de trading de cryptomonnaies.

Ce projet est la suite de projets déjà réalisés par Mr. Pighini, Mr. Toniutt et Mr. SouzaLuz. Il est réalisé dans le cadre des projets de semestre et de Bachelor à HEPIA et a pour objectif d'améliorer le simulateur de trading de cryptomonnaies existant, notamment grâce à des notions de développement logiciel comme la programmation orientée objet. Ces méthodes de développement permettront de modulariser le code pour plusieurs raisons comme une meilleure maintenabilité, un code plus intuitif et de meilleures performances. En plus de la réimplémentation du projet dans un langage orienté objet, les données seront récupérer d'une API et plus d'une base de données. Ce changement de source de données nous permet d'avoir les données les plus récentes et plus d'être restreint à des données fixes sur une certaine durée comme avoir les données de l'année 2022 seulement.

Structure de la documentation

Cahier des charges

- Choisir une API permettant de récupérer les données de cryptomonnaies depuis au moins 5ans.
- Afficher un graphique des bougies d'une cryptomonnaie sur une certaine période
- Créer le simulateur permettant de simuler l'évolution du prix d'une cryptomonnaie toutes les \$x\$ secondes
- Implémenter les différents indicateurs
- Créer et gérer un système de cache pour les données (moins important pour le moment)

Simulateurs de trading de cryptomonnaies existants

Voici certains simulateurs de trading de cryptomonnaies déjà existant : BitGasp : <https://app.bitsgap.com/>
Avatrade : <https://webtrader6.avatrade.com/>

Choix technologiques

Frontend

Pour faire l'affichage d'un graphe pour l'affichage des bougies, j'utilise la librairie lightweight-charts open-source de TradingView. Pour l'instant, je fais un frontend simple HTML-CSS-JS et plus tard je ferai un frontend plus complexe et plus réactif en utilisant soit Angular soit Vue.js ou soit React.

Backend

Java/Spring Boot

Avantages :

- Configuration du projet simplifiée
- Spring Boot utilise Maven ou Gradle pour la gestion des dépendances et de la compatibilité entre elles
- Gestion des objets et de leur cycle de vie
- Plusieurs fonctionnalités simplifiées comme la connexion à une base de données ou l'implémentation d'une API

Désavantages :

- La prise en main peut être compliquée notamment si l'on ne connaît pas les différentes annotations et à quoi elles servent

C#

Avantages :

- Accès au framework .NET

Désavantages :

- Moins de librairies open-source

- Implémentation d'une API plus compliqué qu'avec Spring Boot

Python

Avantages :

- Prise en main facile pour les débutants avec de nombreuses fonctions et librairies déjà implémentées qui ne le sont pas forcément dans d'autres langages
- Implémentation d'une API assez simple grâce à FastAPI ou Flask

Désavantages :

- Langage interprété, moins bonnes performances
- Pas réellement orienté objet

Node.js/Typescript

Avantages :

- Gestionnaire de package NPM

Désavantages :

- Temps de transpilation

Récupération des données

Listes de plusieurs API :

- Binance
- HitBTC
- Kraken
- CryptoCompare
- Poloniex

Pour récupérer les bougies (candles) nécessaires pour pouvoir faire l'interpolation du prix toutes les X secondes, nous récupérerons les bougies à intervalles de 15 minutes depuis l'API de Binance.

La transmission des données depuis le backend vers le frontend se fait via une API dont les routes seront décrites plus tard.

Description des routes de l'API

Lien sur Spring Boot <https://www.ibm.com/topics/java-spring-boot> Lien sur C# <https://dyma.fr/blog/csharp/>