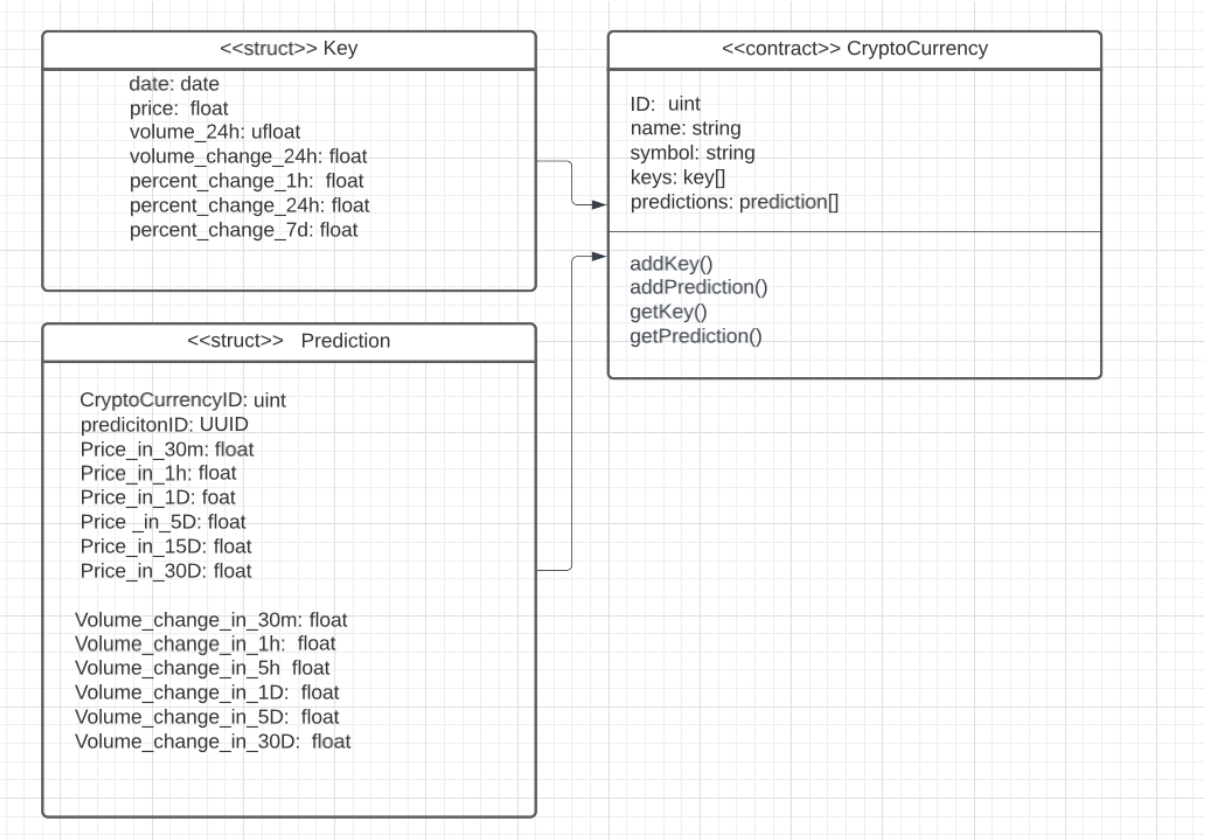
V - La récupération des données :

La récupération des données des crypto-monnaies (toutes les données concernant leurs prix, le volume disponible…) se fait à l'aide d'une API proposée par une application tiers, CoinBase. Cette API fournit toute les informations nécessaires pour que l'IA de prédictions de variations du marché crypto puisse s'améliorer et proposer des prédictions toujours plus précises.

VI- Hébergement des données et des fonctions

Pour ce projet nous prévoyons d'utiliser la blockchain pour héberger l'intégralité des données relatives aux crypto-monnaies ainsi que les prédictions réalisées par notre IA. En effet, le fonctionnement de la blockchain permet à n'importe quel utilisateur de y déployer des données/fonctions à l'intérieur. Cependant, une fois déployer ces données et fonctions ne sont plus modifiables ou deviennent très dur à modifier. C'est pour cela que beaucoup de rigueur sera demandée lors du développement des fonctions pour ne pas risquer de perdre de l'argent en ayant déployé des fonctions fausses.

Prenons pour exemple notre version du déploiement des données des crypto-monnaies ainsi que les prédictions faites par l'IA. Pour illustrer nos propos prenons les parties concernées du diagramme de classe :



Dans cet exemple les classes ayant la mention <<struct>> sont des structures de données ne contenant aucunes fonctions. Dans l'exemple ci dessous la structure key représente les données reçues par l'API de coinbase et la structure prédiction représente les données émises par notre IA. Concernant la classe <<contract>> CryptoCurrency, il s'agit d'un Smart Contract. Les Smart Contract sont à la base de tout le langage solidity car il s'agit de ces derniers qui seront déployés sur la blockchain. On retrouve donc deux tableaux keys[] et predictions[] qui contiennent les données. Ainsi que les méthodes addKey() et addPrediction() pour ajouter des données.

`addKey()`: Permet d'ajouter une key au contrat CryptoCurrency. On retrouvera dans cette méthode une mention pour s'assurer que seules les informations récoltées par l'api puissent être ajoutées.
`addPrediciton()`: Permet d'ajouter une prédiction au contrat CryptoCurrency. On retrouvera dans cette méthode une mention pour s'assurer que seul l'IA de prédiction puisse être ajoutée.

VII -robot de trading et d'alerte

Nous souhaitons mettre en place différents types de robots de trading tous ayant des fonctions différentes. Chaque utilisateur pourra acheter un de ces robots pour que ces derniers fassent des réalises des transactions en fonction des prédictions faites par l'IA. C'est une fois acheté et paramétré que ce robot sera déployé sur la blockchain.

Les robots diffèrent de part leurs comportements. En effet nous mettrons en place trois robots :

Le Bot_Calme

Ce robot pourra réaliser des transactions via les portemonnaies des utilisateurs, cependant il aura un comportement calme, il ne cherchera pas à prendre trop de risque en se fiant à des prédictions trop longues sur la durée. Il cherchera juste à faire des profits avec de très peu risque de perte.

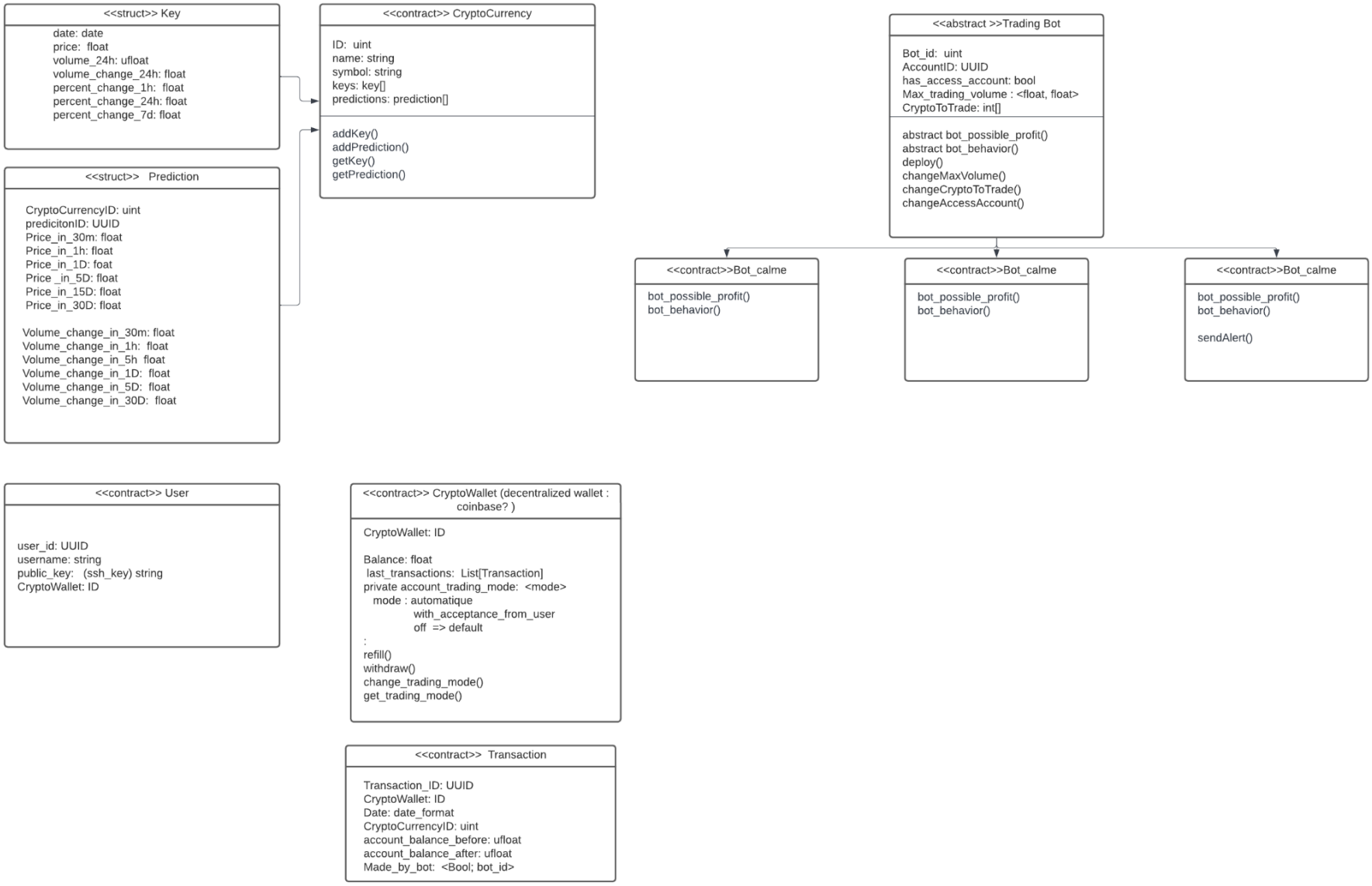
Le Bot_téméraire

Ce robot comme le calme pourra réaliser des transactions via les portemonnaies des utilisateurs, cependant il prendra plus de risque. Résultat dans des meilleurs profits mais aussi des plus grandes chances de pertes.

Le Bot_guide

Ce robot ne pourra pas réaliser lui-même les transactions via les portemonnaies des utilisateurs. Il sera là que pour envoyer des alertes auprès des utilisateurs. L'objectif sera de les guider pour qu'il puisse faire des profits.

VIII- Diagramme de classe



IX - Solidity et Blockchain

Dans solidity nous créons nos classes et les structures dans le dossier src/solidity/Contracts. Dans le dossier src/solidity/script nous retrouvons les script qui nous permettent de déployer contre une certaines somme de Ethereum, les contrats préalablement définis.