CrypTrade

Investir dans les cryptos n'a jamais été aussi facile ! Le Bitcoin, l'Ethereum, le Monero, tous à portée de main !

Lyon 2021
EPITA

BUTHOD Victor
victor.buthod@epita.fr

MASSON Jacques
jacques.masson@epita.fr

MAYER Julien
julien.mayer@epita.fr

FAURE Joris
joris.faure@epita.fr

Table des matières

Première partie

Introduction

CrypTrade est un logiciel de trading de crypto-monnaie qui dans un premier temps, récupére la valeur réelle de différentes crypto-monnaies à partir d'une API (sur ce site par exemple), afin de créer un simulateur conforme au cours réel des cryptos-monnaies. Vous pouvez acheter et vendre ces monnaies manuellement, et prochainement vous pourrez le faire automatiquement via une intelligence artificielle intégrée.

1 Qu'est-ce qu'une crypto-monnaie ?

À l'inverse des monnaies "physiques" comme l'Euro, la Livre ou encore le Dollar, émises et régulées par les gouvernements, une crypto-monnaie est une monnaie virtuelle, dont l'émission et l'évolution ne dépendent pas d'un état ou d'une banque centrale mais qui repose sur l'ensemble de ses utilisateurs.
On peut citer notamment le Bitcoin, mais aussi l'Etherum, le Monero, le Lite Coin, le Doge Coin, etc.

Toutes les crypto-monnaies reposent sur ce que l'on appelle la Blockchain (littéralement la "chaine de blocs").
Pour comprendre ce dont il s'agit, il faut vous représenter la Blockchain comme un immense livre de comptes dans lequel sont inscrites toutes les informations des transactions de crypto-monnaies, celles-ci ne sont alors pas vérifiées par une banque mais par tout un réseau d'utilisateurs, les mineurs.
La Blockchain contient une trace de l'intégralité des échanges réalisés jusqu'ici. Elle est non seulement indestructible, mais surtout consultable à tout moment par n'importe qui.

2 Principe du trading

La stratégie principale du trading est la suivante :
L'objectif est d'acheter une monnaie avant que la valeur de celle-ci augmente et de la vendre juste avant qu'elle ne baisse.

Avec l'avènement du trading algorithmique sur les places financières mondiales, de nouvelles technologies voient le jour dans les institutions financières. En effet, les programmes informatiques peuvent être bien plus efficaces que n'importe quel trader pour analyser les données et réagir au bon moment.

3 Notre algorithme

C'est là que commence la partie algorithmique du projet. Un bot sera intégré au logiciel, nous convertirons différentes stratégies de trading en algorithme afin que notre bot achète ou vende des monnaies à des moments optimisés.

En plus de l'algorithme de base, nous implémenterons plusieurs fonctionnalités additionnelles qui permettent d'optimiser les transactions du bot de la même manière que le ferait un vrai trader de Wall Street.

Deuxième partie
L'équipe

4 Victor Buthod

Pour cette première période de soutenance, en tant que chef du groupe 667 investments, j'ai encadré la réalisation de toutes les parties du projet CrypTrade.

Afin d'avoir une base pour commencer la réalisation du Bot, j'ai commencé par faire une interface graphique avec le logiciel Glade, qui aura par la suite été améliorée par Jacques et Joris.

Une fois la maquette de l'interface terminée, j'ai créé un programme pour récupérer les valeurs des monnaies en temps réel. Pour cela, j'ai utilisé un de nos ancien TP d'informatique pratique qui récupérait le contenu des sites en HTML. Le principe était donc de récupérer le contenu d'un site de cryptomonnaies (tel que cryptowat) en les enregistrant dans un fichier texte, pour pouvoir ensuite rechercher les données qui nous intéressaient afin de les mettre dans une structure monnaie contenant des variables tel que le nom de la crypto-monnaie, sa valeur, son évolution depuis les dernières 24 heures, et bien plus. J'ai ensuite découvert une stratégie bien plus efficace qui consiste à utiliser une API pour que les données soient enregistrées de manière plus condensée, après un peu de recherche, j'ai trouve l'API coin cap qui convenait parfaitement à notre projet, car les informations sont compactées de manière optimisée (exemple ici avec l'API de la monnaie bitcoin).

J'ai finalement utilisé la librairie "curl" pour récupérer les données du site plus efficacement qu'avec notre ancien TP de c, car cela prenait trop de temps et de mémoire à notre programme. Les données récupérées et enregistrées dans un fichier texte, Julien et moi avons fait un programme permettant d'y chercher les valeurs intéressantes, et les enregistrer dans une structure, pour pouvoir les réutiliser plus tard dans les autres parties du projet.

Après que Joris et Jacques aient relié les données des structures aux boutons de l'interface graphique de Glade, je les ai aidé à créer les fonctions pour acheter et vendre (Buy/Sell sur Glade) des crypto-monnaies.

J'ai ensuite du debugger avec valgrind le fichier fetcher.c qui nous permet de récupérer les valeurs des cryptomonnaies, car beaucoup d'allocations de mémoire sont faites, et après une longue utilisation de CrypTrade, des fautes de segmentation arrêtaient notre programme.

Il nous fallait ensuite un graphique pour avoir une interface utilisateur permettant de mieux comprendre la fluctuation du prix des monnaies, mais intégrer un graphique qui s'actualise en temps réel sur Glade n'est pas chose facile, car il n'y a pas d'outil sur le logiciel permettant de le faire directement, nous pouvons seulement afficher une image. J'ai donc fait des recherches pour créer un graphique lié avec Glade, mais je n'ai trouvé que des programmes utilisant des fonctions en python, j'ai donc décidé de faire un programme en ce qui ne faisait que créer une image de graphique en PNG, j'ai trouvé sur Github un programme en ce qui convertit deux listes de coordonnées en graphique. Cependant le graphique généré par le programme requiert d'avoir un historique des valeurs (pour l'abscisse) en fonction du temps (pour l'ordonnée), ce que nous n'avons pas encore implémenté. Je crée donc, pour l'instant, une image avec un point seul qui symbolise la valeur du bitcoin au moment du lancement du programme.

J'adore travailler sur ce projet, car même si le c'est un language que je trouve difficile, je sens que je m'améliore vite et je découvre plein de choses en allant chercher des informations pour avancer par moi-même.

5 Joris Faure

Durant cette première phase de développement, j'étais responsable de la partie Interface Graphique de notre application.
Après avoir réalisé la maquette de l'interface avec Victor, j'ai implémenté l'interface finale avec Glade, puis je l'ai relié au fichier UI.c avec la bibliothèque GTK. Pour pouvoir développer de façon optimisée avec Jacques, j'ai implémenté le squelette du fichier UI.c avec l'initialisation de l'interface, la récupération de tous les widgets ainsi que l'organisation des différentes fonctions reliées aux widgets. Tous les widgets étaient donc placés et reliés à leurs fonctions principales.
Après avoir posé les bases du code, j'ai essayé (avec l'aide de l'algorithme de récupération des données des crypto-monnaies "fetcher.c" de Victor et la création de la structure Money de Julien) d'afficher la valeur actuelle d'une crypto-monnaie lorsque l'on la choisissait via les boutons de l'interface. Seulement une erreur faisait que la structure Money n'était pas reconnue et empêchait la récupération du prix des monnaies. Victor, Julien et Jacques ont alors repris leurs fonctions ainsi que le makefile afin de permettre la récupération du prix, et j'ai ainsi pu

implémenter l'affichage des valeurs des crypto-monnaies. Par la suite, cet affichage a été amélioré par Victor afin de permettre une actualisation automatique de ces valeurs sans obligatoirement recliquer sur les boutons.
Un autre problème s'est aussi posé, c'est que Glade, bien que le Bitcoin soit de base sélectionnée, n'affichait pas la valeur du bit coin, il fallait de nouveau cliquer sur le bouton "bitcoin". J'ai donc dû initialiser la fenêtre glade avec de base la récupération et l'affichage de la valeur du Bitcoin. Après avoir réglé le problème de récupération des valeurs des monnaies, on a pu se concentrer sur les boutons Buy et Sell.
J'ai donc implémenté le bouton Buy qui permet d'acheter les monnaies voulues et les affiche dans le stock de l'utilisateur. Cet affichage a par la suite été amélioré par Jacques en reliant dynamiquement le stock de crypto-monnaie et leurs équivalences en dollars, ainsi que la possibilité de garder les monnaies préalablement achetées. Enfin la fonctionnalité de vente et de portefeuille on était implémentée par Jacques et Victor. Je suis content du travail fourni lors de cette première soutenance car l'interface graphique est presque finalisée. Il manque surtout l'affichage du graphique des crypto-monnaies au centre de l'interface. Étant déjà renseigné sur la manipulation des images avec GTK, Jacques et moi afficherons en fonction des monnaies choisis, les graphiques en .Png généré par Victor.

6 Julien Mayer

Pour cette première soutenance, je me suis occupé de la partie "récupération des données" avec Victor. Le but de cette étape était de pouvoir se créer une base de données sur les crypto-monnaies. Elle permettra de faire fonctionner le simulateur de trading et l'analyse par le Bot.
Pour que notre simulateur marche au mieux, il faut que les données des monnaies soient les plus récentes possible et faciles à récupérer.Alors on utilise l'API dun site CoinCap pour les télécharger via curl afin de les enregistrer sous forme de fichier texte.Il nous fallait un moyen efficace pour organiser et manipuler ces informations. Alors mon rôle était d'implémenter une structure et des fonctions pour ces données permettant de faciliter la suite du projet.J'ai déclaré une structure Money, compatible avec toutes les crypto-monnaies, possédant des variables pour stocker les informations telles que son nom, son symbole, son rang, son prix actuel ...

Une fois les structures mises en place j'ai travaillait sur une méthode pour créer la structure d'une crypto-monnaie depuis son fichier texte télécharger précédemment. Pour ça, je récupère le texte du document pour le parcourir afin d'y prélever les informations importantes. J'utilise la détection de certains caractères dans ce texte pour trouver et isoler les informations de la crypto-monnaie. Pour bien construire la structure, ses différentes variables sont indexées par l'ordre de rencontre des informations dans le texte.
Tout ce processus est effectué à l'appel de la fonction *get_struct* avec un nom

de crypto-monnaie en argument. La fonction renvoie directement la structure de la monnaie.
J'ai aussi intégré la fonction *get_ strc_ list* prenant une liste de monnaies et qui renvoie une liste de structure. Celle-ci nous permettra de facilement actualiser plusieurs valeurs de structures monnaies.

7 Jacques Masson

Pour cette première soutenance, j'ai été responsable du site et suppléant sur l'interface graphique. Pour le site, ce dernier a été mis à jour malgré encore quelque fonctionnalité manquante, la plupart des pages commencent à être bien faites. Il ne nous manque plus que la page des réseaux, mais vu que le site n'est pas le plus important j'ai préféré aider Joris sur l'interface graphique. Pour cette dernière, Joris ayant créé le squelette de l'interface ce fut plus simple après de m'expliquer comment marcher Glade.
Une fois mes premiers pas sur Glade fait, j'ai pu ajouter et améliorer plusieurs choses à l'interface graphique. On a pu créer les boutons Buy/Sell et actualiser le wallet de l'utilisateur.
Ces deux premières fonctions permettent à l'utilisateur d'acheter et de vendre sa crypto-monnaie, les problèmes étant que certaines transactions ne marchent pas encore très bien, mais dans l'ensemble la plupart se passent comme prévu, aussi nous avons géré le cas des changements de monnaie, si l'utilisateur veut acheter ou vendre plusieurs monnaies. Grâce à la récupération de données sur les cryptomonnaies faites par Julien et Victor, nous avons pu avec Joris afficher les bonnes valeurs de la monnaie et les actualiser sur l'interface. Une fois cela fait nous avons pu créer les différentes actions que va réaliser l'utilisateur. Ainsi, ce dernier peut simuler l'achat et la vente de la crypto-monnaie à une valeur toujours actualiser et la plus proche de sa valeur réelle. En même temps que nous avons implementer les options Buy et Sell nous avons dû utiliser le wallet pour mettre ce dernier à jour à chaque transaction, ainsi nous avons pu améliorer les fonctions d'achats pour permettre à l'utilisateur de cumuler ses achats et de pouvoir utiliser plusieurs crypto-monnaies en même temps.
Je suis satisfait de la production que nous avons réalisé jusqu'à présent car je partais en ne connaissant rien à Glade et maintenant, je pense que nous aurons fini très bientôt car nous avons déjà plein d'idées pour corriger les problèmes de transactions et pour afficher un graph qui va s'actualiser au fil du temps pour montrer les courbes d'évolution des monnaies.

Troisième partie
Notre projet

8 Récupération des données

8.1 API

Pour récupérer les valeurs des crypto-monnaies en temps réel, nous avons utilisé l'API de coincap.io qui est simplement un site sur lequel toutes les valeurs des cryptomonnaies sont répertoriées comme sur l'image ci-dessous :


```
{"data":
{"id":"bitcoin","rank":"1","symbol":"BTC","name":"Bitcoin","supply":"18672456.0000000000000000",
"maxSupply":"21000000.0000000000000000","marketCapUsd":"1074833911484.8342762639064136","volumeU
sd24Hr":"24664097018.8250757255234417","priceUsd":"57562.5355060327509281","changePercent24Hr":"
-1.9084793051051737","vwap24Hr":"58073.5010790271633669","explorer":"https://blockchain.info/"},
"timestamp":1617789448137}
```


Elle fonctionne de la façon suivante ; il nous suffit d'ajouter le nom de la crypto-monnaie désirée après l'URL *https://api.coincap.io/v2/assets/*, par exemple sur l'image ci-dessus, l'URL est *https://api.coincap.io/v2/assets/bitcoin*. Le nom de la crypto-monnaie est donc passé à la fonction *update_value*, puis concaténé a une liste de caractère contenant l'URL de l'API. Par la suite, on utilise la librairie curl pour télécharger le contenu du site, qu'on écrit dans un fichier grâce à la librairie "string h".

8.2 Structure

Une fois les fichiers textes des monnaies souhaitées créés, on récupère les informations de chaque crypto-monnaies en parcourant son ficher.
Ces données seront organisées dans une structure Money où chaque variable correspond à une information sur la crypto-monnaie.
Pour détecter les valeurs importantes dans le texte, on cherche les endroits où les caractères : et " se suivent.
Le champ à récupérer se trouve juste après, et jusqu'au prochain caractère ". On utilise la fonction atof pour convertir la chaîne de caractères isolée du texte en float, CoinCap utilise un . pour marquer les décimales, alors que cette fonction ne reconnaît que les virgules. Pour régler ce problème, lors d'un parcours du texte, on remplace les points par des virgules.
Une fois la donnée isolée du reste du texte, on l'affecte à la variable correspondante dans la structure Money créée auparavant.
Pour savoir dans quelle variable doit aller à quelle donnée, on utilise un index qui s'augmente dès qu'une des variables est affectée. De cette manière, on peut organiser les valeurs dans la structure.

Ce processus est effectué à l'appel de la fonction *get_strc* avec un nom de monnaie en argument.
On peut aussi récupérer une liste de structure en appelant la fonction *get_strc_list* avec une liste de noms de monnaies en argument.

```
----------------------------------------------------
id = bitcoin
rank = 1
symbol = 'BTC'
name = 'Bitcoin'
supply = 18672456,000000
maxSupply = 21000000,000000
marketCapUsd = '1079775002624,000000'
volumeUsd24Hr = '22537943040,000000'
priceUsd = 57827,156250
changePercent24Hr = '-1,278040'
vwap24Hr = '58108,023438'
----------------------------------------------------
```

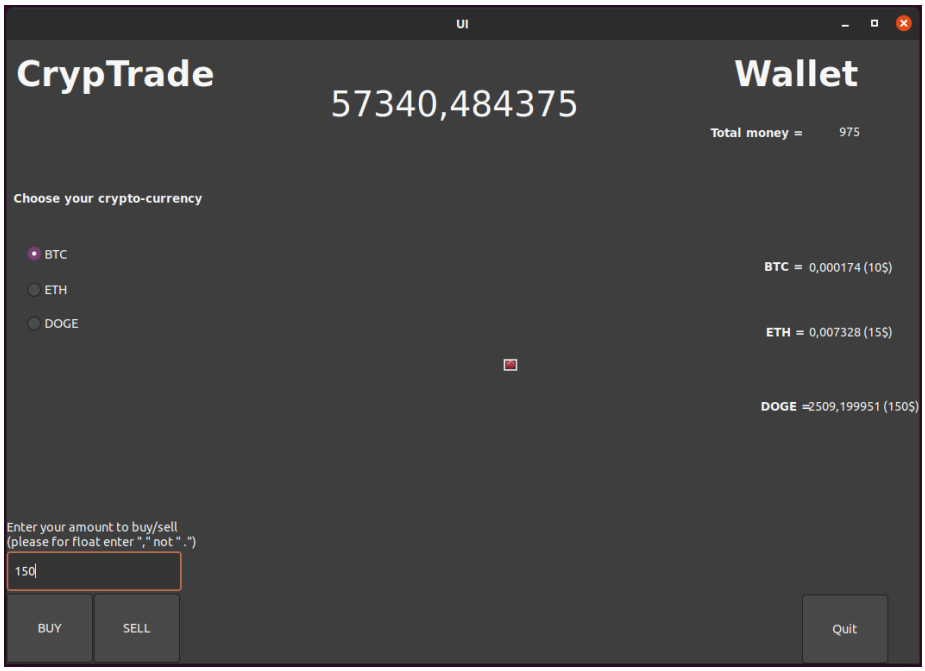
Example d'une structure avec le bitcoin

9 Interface du Simulateur

9.1 Graphique

Pour réaliser l'interface graphique de notre simulateur de trading, nous avons utilisé Glade basé sur la bibliothèque GTK. Glade prend en charge toute la partie gestion et génération de l'interface ce qui nous a permis de nous concentrer sur la partie code "utile". Nous déclarons donc notre main dans UI.c qui initialisera notre fenêtre GTK, et stockera dans des variables globales tous les widgets de l'interface (les widgets sont les différents éléments de l'interface graphique tels que les boutons, l'espace d'entrée utilisateur, les labels etc.). De cette façon nous avons la base de l'interface, ce qui nous permettra de manipuler aisément chaque élément graphique.

Nous avons choisi de faire une interface claire et intuitive qui se présente comme suit :



1. Gauche : Choix des crypto-monnaies
2. Bas/Gauche : Entrée utilisateur (en $) et boutons Acheter/Vendre
3. Haut : Valeur en temps réel de la monnaie choisis
4. Milieu :Futur graphique du cours de la monnaie sélectionnée
5. Droit/Haut : Valeur atuelle du porte-monnaie (en $)
6. Droit : Quantité de monnaie possédé par l'utilisateur
7. Droit/Bas : Bouton Quitter

9.2 Trading Simulateur

9.2.1 Récupération des données dans le simulateur

Tout d'abord un thread sera créé et exécutera une boucle infinie qui vérifie constamment sur quelle monnaie on se trouve grâce à la variable globale *on_money*, et met à jour toutes les 5 secondes l'affichage de la valeur de la monnaie sélectionnée grâce à des fonctions intermédiaires. Ces fonctions récupèrent, grâce à la structure Money, la valeur actuelle d'une crypto-monnaie en temps voulu. Pour ce faire, on récupère l'attribut *priceUSD* de la structure Money. Ensuite, on va inscrire la valeur dans le label souhaité (voir le numéro 3 sur la photo de l'interface graphique).

```
1  void on_btc_graph_button_toggled(){
2    on_money = 0;
3    if (1){
4      struct Money *strc = get_strc("bitcoin");
5      float val = strc->priceUsd;
6      char array[100];
7      sprintf(array, "%f", val);
8      gtk_label_set_text(GTK_LABEL(value_label), (gchar*)array);
9      }
10 }
```

Si l'utilisateur change de bouton, il exécute la fonction intermédiaire qui lui est associée et qui changera la variable *on_money* afin de permettre à la boucle qui actualise les valeurs d'afficher uniquement la valeur de la monnaie sélectionnée.

9.2.2 Buy/Sell

Chaque fois que l'utilisateur clique sur le bouton BUY, cela lance la fonction *on_buy_button_clicked* qui exécute selon la monnaie choisie la fonction *change_crypt_amount* qui prend en paramètre le nom de la monnaie sélectionnée. Cette dernière fonction récupère la valeur entrée par l'utilisateur sur l'interface grâce à la fonction *on_value_entry_changed*, ainsi que la valeur des monnaies en temps réel. Par la suite, le nombre de crypto-monnaies achetées ainsi que leur équivalence en $ est affiché à droite de l'interface.
Le bouton SELL fonctionne exactement de la même manière que buy, mais la valeur flottante *pos*, qui est initialisée au début du fichier, est changée à "-1", alors qu'au début de BUY, elle est mise à "1".
Dans la fonction "change_crypt_amount", la nouvelle valeur du porte-monnaie et celle des valeurs des crypto-monnaies achetées sont calculées en fonction de la variable "temp", qui est le nombre entré par l'utilisateur.

Cette variable est multipliée par *pos*, quand l'utilisateur utilise le bouton BUY, elle est multipliée par 1, et donc ne change pas. Alors que lors de l'utilisation du bouton SELL, la valeur est multipliée par -1, et devient négative. Elle ajoute donc au porte-monnaie au lieu d'enlever, et retire à la valeur des monnaies, au lieu d'ajouter.

9.2.3 Wallet

L'option wallet va permettre à l'utilisateur de voir l'argent qu'il a gagné et perdu au fil de sa simulation. Sur la photo de l'interface c'est la partie en haut à droite. L'utilisateur commence donc avec une valeur définie, pour l'instant cette dernière est fixée à 1000 dollars, ce qui permettra à l'utilisateur de faire des transactions au départ. Une fois qu'un montant est passé par le biais de *on_ value_ entry_ change*, l'utilisateur doit entrer le type de transaction qu'il souhaite faire. Le wallet sera incrémenté selon cette dernière.
Cette première version du wallet nous permet donc de tester les boutons BUY et SELL grâce à la valeur définie de base, mais il n'est pas encore complet. Il faudrait qu'il puisse augmenter ou baisser sa valeur selon le cours des monnaies actuelles, mais pour ce faire, nous avons besoin de la partie finance sur laquelle nous n'avons pas encore travaillé.

9.2.4 Problèmes rencontrés

Nous avons eu plusieurs problèmes liés à l'interface graphique. Le fait que nous manipulions des grosses structures de données dans l'interface nous a causé certains problèmes de mémoires. De plus, nous avons encore des soucis pour lier le graphique à l'interface. Nous avons eu du mal avec l'actualisation en temps réel des valeurs, cela n'aurait aucun intérêt de simuler l'achat et la revente de crypto-monnaies sur des valeurs obsolètes.
Enfin, pour afficher un graphique, nous ne trouvions pas de solutions pour le en créer uniquement en C, c'est pour cela qu'une partie de notre code était en python. Nous pensions utiliser les deux languages, mais nous sommes arrivés à trouver des librairies et des méthodes nous permettant de faire ce que nous voulions uniquement en C. Par ailleurs, nous n'avons pas encore eu le temps de les implémenter correctement.

Quatrième partie
Site internet et Réseaux

10 Site internet

Pour la réalisation du site internet nous allons coder en HTML car cela nous permet de créer un site assez simple que nous compléterons avec du code CSS pour embellir le résultat. Le site sera composé de plusieurs pages sur lesquels l'utilisateur pourra se déplacer.
La première page sera l'accueil, cette page regroupe des photos de notre projet ainsi qu'une partie avec les actualités nouvelles et anciennes. Comme deuxième page, nous avons la page Groupe, cette page est dédiée à une courte présentation du groupe et des personnes qui le compose pour réaliser le projet. La page réseaux sociaux, elle servira à donner tous les liens vers nos pages Twitter, Instagram, et autres si nous en trouvons l'utilité.
La page FAQ, elle, est faite pour permettre à nous, les développeurs, d'interagir et de répondre aux questions des utilisateurs, et ainsi leur permettre une meilleure expérience de l'application. La page suivante est la page des liens utiles, dans cette page nous avons inscrit nos différents emails, et aussi les différents des ressources que nous avons utilisées.
Il y aura la page chronologie de la réalisation où nous raconterons toutes les étapes de la réalisation du projet de ses débuts jusqu'à sa finalité.Enfin il y aura une page de téléchargement du logiciel où l'utilisateur pourra télécharger le logiciel sous plusieurs versions différentes afin qu'il puisse choisir la version qu'il désire. D'autres pages arriveront si nous pensons qu'elle est nécessaire pour l'utilisateur et ses demandes, notamment une page sur le projet en lui-même, c'est-à-dire nos motivations à le faire, ce qui le rend unique par rapport aux autres, et toutes les étapes de sa conception.

Le site actuel de CrypTrade

11 Réseau sociaux

Pour la promotion de notre application, nous avons décidé de lui créer des pages dédiées sur les différents réseaux sociaux tels que Instagram ou encore Twitter. Cela permettra aux utilisateurs de suivre l'avancement de notre application. Aussi ces pages nous serviront à interagir avec les différents utilisateurs, recevoir leurs critiques afin d'améliorer l'application, ou encore de créer des tutoriels pour les débutants dans l'application..

Cinquième partie
Répartition des charges

	Répartition des tâches				
	Interface graphique	**Stratégie du bot**	**Récupération des données**	**Fonctionnalités additionnelles**	**Le site**
Joris	Resp.	Supp.		Supp.	
Victor	Supp.	Supp.	Resp.	Resp.	
Julien		Resp.	Supp.	Supp.	
Jacques	Supp.	Supp.			Resp.

Sixième partie
Conclusion

En conclusion pour ce premier rapport de soutenance, nous avons bien avancé ce projet que nous voulons tous mener à bien, le fait que l'équipe soit aussi soudée permet les interactions entre les membres plus efficaces. Pour résumer l'avancée du projet, nous avons fini la récupération de données, ce qui nous permet d'être à jour sur les valeurs des monnaies en temps réel. Pour ce qui est de l'interface, quelques problèmes subsistent dans l'optimisation des fonctions et la gestion de la mémoire, la structure du code, ainsi que quelques améliorations qui apparaîtront au cours du temps. Actuellement, nos plus gros problèmes sont nos erreurs de segmentation. Pour ce qui est de l'avancement général du projet, nous avons respecté ce que nous avions prévu de faire pour cette première soutenance, nous ne sommes donc pas en retard quant à l'avenir du projet. Il nous reste encore à faire tous les algorithmes pour les différentes stratégies que CrypTrade sera capable d'adopter.

Septième partie
Annexes

lien vers Openclassroom : `https://openclassrooms.com/fr/`
lien vers Youtube : `https://www.youtube.com/`
lien vers StackOverflow :`https://stackoverflow.com/`
lien vers Gold Price : `https://goldprice.org/`