Techniques de développement logiciel

Rapport du projet

Dashboard financier

Antton D'ELISSAGARAY
Isabel FERREIRA FORMOSO DA SILVA
Murilo COSTA CAMPOS DE MOURA

Janvier / 2022

École des Ponts ParisTech

Table de matières

Objectif initial

L'objectif initial de ce projet était de créer un tableau de bord en ligne dans lequel un utilisateur accederait aux données financières de certaines entreprises afin de montrer l'évolution des prix de leurs actions, fournir des recommandations d'achat, composer un portefeuille, donner la valeur prédictive de l'action et afficher des graphiques avec des prévisions de prix.

Le groupe a essentiellement divisé les tâches de chacun en trois parties :

1. Code Python avec les fonctions pour accéder aux bases de données financières en ligne et donner des résultats financiers;

2. Codes HTML/CSS pour la partie visuel du site web;

3. Codes spécifiques pour faire le login, enregistrement et logout et enregistrer les favoris.

Plusieurs outils ont été utilisé dans le projet :

- Flask : un micro framework open-source de développement web en Python qui a joué le rôle du serveur ;

- HTML : pour créer la structure des pages web ;

- Bootstrap : une source de codes CSS avec des templates prédéfinis qui ont servi à rendre le site web plus organisé et visuel.

Réalisation effective

Le logiciel actuel est un site web qui permet l'utilisateur de :

- se connecter à un site avec une interface graphique élaborée ;

- chercher par un ticker (l'abréviation utilisée pour identifier les actions cotées en bourse d'une entreprise sur un marché boursier) ;

- visualiser sur une nouvelle page les courbes des actions d'une certaine entreprise ;

- faire l'enregistrement sur le site, faire le login et faire des recherches en mode connecté.

Pour la semaine prochaine, nous reste à faire :

- sauvegarder dans une base de données une certaine entreprise comme favorite et montrer cette information sur la page de profil de l'utilisateur.

En comparant l'état actuel avec l'objectif initial, le groupe n'as pas eu le temps de mettre en oeuvre les fonctionnalités suivantes :

- recommandations d'achat ;

- composition d'un portefeuille pour l' utilisateur ;

- des graphiques avec des prévisions de prix.

- utilisation du module Angular pour le développement web.

Focus sur un point technique

1. Fonction Login

Nous allons à présent nous attarder sur la partie login. Nous ne l'avions pas identifié, au début, comme un point technique de notre logiciel. Néanmoins elle s'est avérée être un point crucial qui nous a demandé plus de temps que ce que nous imaginions. Nous nous sommes répartis la tâche. D'un côté la partie visuelle, à l'aide de HTML, CSS et Bootstrap. De l'autre la partie backend.

Nous avons pris du temps à prendre l'outil flask en main. C'est un module de Python qui permet de faire du développement web. Le début de cette partie nous a demandé de regarder beaucoup de tutoriels internet et de nous documenter. Ce temps de documentation, bien qu'il ait été très fructueux, a retardé le début d'écriture du code.

Pour mettre en œuvre une fonctionnalité login, nous avons dû créer une base de données User, qui prenait en compte l'adresse mail de l'utilisateur, son nom d'utilisateur et un mot de passe. Nous avons aussi ajouté une colonne "favoris" afin d'enregistrer dans son profil ses entreprises préférées. Lors de sa première connexion nous avons mis les favoris à 'null'. Ensuite il a fallu créer deux classes RegisterForms et LoginForms qui permettent à l'utilisateur de s'enregistrer et de se connecter à l'interface. A noter, il est possible d'utiliser le site sans être ni enregistré ni connecté. Pour pouvoir utiliser toutes ces fonctionnalités, nous avons utilisé les modules SQLAlchemy ou LoginManager de flask. Nous avons aussi créé un SecretKey afin de sécuriser les connexions au site. Nous avons aussi fait en sorte que les mots de passe soient hachés afin de protéger les données de l'utilisateur en cas de piratage. Enfin nous avons permis à l'utilisateur de se déconnecter. Nous avons aussi utilisé le module flash de flask afin d'indiquer à l'utilisateur qu'il se trompe de mot de passe ou de username lorsqu'il se connecte.

En effectuant tout cela nous avons eu quelques problèmes. Nous avons par exemple eu des problèmes d'affichage. A l'heure où nous écrivons ce rapport, l'affichage 'mot de passe invalide' fonctionne, en revanche (alors que le code est similaire) l'affichage dans la page Register 'email adresse déjà utilisée' ne fonctionne pas même si le logiciel reconnaît que l'utilisateur est déjà existant. Nous avons aussi des problèmes de base de données. Pour l'instant, la base de données des nouveaux utilisateurs n'est enregistrée que lorsque que le site fonctionne. Dès que l'on quitte le site tout s'efface, il faut s'enregistrer de nouveau si l'on veut se créer un profil lors d'une nouvelle connexion. Nous avons identifié le problème, il vient de l'utilisation de la fonction create_all() dans le module SQLAlchemy, néanmoins nous ne l'avons pas encore résolu.

Pour pouvoir résoudre les problèmes d'affichage et les soucis de communication entre le frontend et le backend, nous avons testé systématiquement les changements mis en œuvre lors du développement. Etant donné que nous n'avions pas un nombre démesuré de ligne de code nous avons effectué des test unitaire à la main. Pour ne prendre qu'un exemple, lorsqu'il s'agissait de vérifier l'application des fonction d'affichage ('mot de passe invalide', 'cette adresse mail est déjà utilisée') nous avons fait des tests en rentrant le mauvais mot de passe avec une bonne adresse mail lors de la phase Login. Nous avons utilisé la fonction 'inspecter' (clique droit sur la page web), qui nous a permis de revenir à la source HTML afin d'identifier de potentielles erreurs dans le script HTML. Au fur et à mesure de ces tests, nous avons pu améliorer notre programme.

La prochaine étape majeure réside dans la bonne utilisation des base de données afin que l'utilisateur puisse ajouter des favoris et qu'il reste enregistré même après une connexion. Il nous faudra aussi régler les problèmes d'affichage lors de la phase d'enregistrement.

2. La valorisation des actions

Pour calculer le prix d'achat recommandé, nous avons utilisé la méthode de valorisation appelée *Discounted Cash Flows,* (flux de trésorerie actualisé en français).

Selon ce modèle, la valeur réelle d'une entreprise est une fonction des flux futurs de trésorerie disponible aux actionnaires, actualisé par un certain taux. Mathématiquement, le modèle peut être décrit par la formule suivante:

$Val = \sum_{n=1}^{\infty} \frac{FCF_n}{1+r}$, où Val est la valeur d'entreprise, FCF le flux de trésorerie libre et r le taux d'actualisation.

Cependant, Val correspond à la valeur de tout le business et nous voulons le prix d'une seule action. Donc, le prix individuel sera calculé de la façon suivante:

$$Prix = \frac{Val - Dette + Trésorerie}{nb\ d'actions}$$

Les valeurs de FCF_1, $Dette$, $Trésorerie$, et $nb\ d'actions$ sont des valeurs qui se trouvent aux états financiers de l'entreprise. Pour les entreprises cotées en bourse, les comptes de résultat sont des documents publics et facilement trouvés sur l'internet. Pour notre programme, nous avons automatisé la recherche des documents des résultats avec la librairie *yfinance* (disponible sur https://github.com/ranaroussi/yfinance). Cette librairie fournit une interface amicale pour les calculs en définissant la classe Ticker, avec laquelle nous avons travaillé. Au-delà d'accéder aux données de Yahoo directement, qui sont actualisées continuellement, la librairie organise dans des dictionnaires au sein de la classe Ticker ces documents financiers.

Par contre, le taux d'actualisation r et le flux de trésorerie futur FCF_n ne sont pas donnés et doivent être estimés. Pour simplifier le programme, nous partons de l'hypothèse

que r est une simples fonction de β, la covariance entre le prix de l'action et le prix moyen du marché. Pour l'estimation des FCF_n, nous avons aussi pris l'hypothèse simplificatrice que la croissance futur est une fonction de EPS, *earning per share*. Les valeurs de β et EPS sont cherchés sur le site https://finviz.com/, par la libraire *Beautiful Soup,* ce qui a fini par limiter la fonctionnalité du programme, vu que cette base ne fournit pas d'informations sur toutes les entreprises qui seraient disponibles sur Yahoo finance.

3. Structure des pages web

Comme dit précédemment, on a importé le module Flask dans le code Python, ce qui nous a permis de créer facilement le site web. Flask est basé sur la boîte à outils Werkzeug WSGI et la bibliothèque Jinja2. WSGI est la spécification d'une interface commune entre les serveurs Web et les applications, et implémente des fonctions comme les requêtes par exemple. Jinja2 est un système de modèles Web qui combine un modèle avec une source de données spécifique pour afficher une page Web. Cela nous permet de passer des variables Python aux modèles HTML.

Étant dans le dossier où on va mettre le fichier de l'application .py, il a fallu avant créer un environnement de développement virtuel, qui en Python s'appelle "venv". Cet environnement virtuel est isolé, ce qui nous a permis d'installer des dépendances uniquement pour le projet sur lequel nous travaillions. Le code suivant indique la structure minimale pour créer un serveur Flask dans Python :

```
from flask import Flask

app = Flask(__name__)


@app.route('/')
def hello():
    return 'Hello, World!'
```

Après cela, on a créé des routes (@app.route('/nom_de_la_page')) pour directionner à chaque page web HTML, après la réalisation d'une instruction spécifique sur le code Python. L'architecture de la côté web est la suivante :

- HOME (la page initiale avec le champs de recherche et les boutons pour faire le login et l'enregistrement; dont le layout est étendu à LOGIN, à REGISTER et à DASHBOARD)
- DASHBOARD (page similaire à HOME, mais est utilisé quand l'utiliser a fait login)
- LOGIN (la page pour faire login)

- REGISTER (la page pour s'enregistrer)
- BASE (c'est uniquement un fichier dont le layout est étendu à PAGE_TICKER et à PROFILE)
- PAGE_TICKER (la page avec les informations financières de chaque entreprise)
- PROFILE (page de profil de l'utilisateur)

Par rapport aux layouts, les styles des items des pages (textes, containers, formulaires, images, etc), qui ont été trouvés en Bootstrap, ont été ajoutés dans quelques fichiers .CSS, qui étaient référencés par les fichiers .HTML.

Retour d'expérience

Antton d'Elissagaray :

Je me suis occupé de la partie Login et gestion de la base de données. Le début de ce projet a été particulièrement difficile. J'ai été confronté à beaucoup de soucis d'installation. Pour pouvoir utiliser le module flask et le module angular (que nous n'avons finalement pas utilisé…) j'ai passé deux semaines à résoudre des problèmes d'installation. J'ai même dû changer d'éditeur, je suis passé de Pyzo, vraiment inadapté au travail demandé, à Visual Studio. Cela m'a demandé une bonne semaine pour vraiment le prendre en main, n'étant pas très au fait des fonctionnalités de ce nouvel éditeur (A noter que Pycharm ne fonctionne pas sur mon PC….). J'ai aussi appris à me servir des différentes commandes de mon ordinateur, commande que je n'avais jamais utilisée auparavant. Il a fallu que j'apprenne les différents types de commande (Commande Windows, Powershell Windows, Anaconda prompt Powershell…) car les invit de commandes qui m'étaient données pour les installations des modules ne fonctionnait pas sur toutes ou n'était parfois adaptée qu'à Linux. Durant ces semaines d'installation, j'ai malgré tout pu solliciter l'aide de notre tuteur par visio pour m'aider à débloquer certaines de mes erreurs. Même si j'ai eu l'impression de perdre énormément de temps durant cette phase, j'ai appris beaucoup de choses très utiles, elle m'a aussi permis de me remettre à niveau sur des fondamentaux dont je n'avais pas connaissance.

Pour l'écriture à proprement dite j'ai passé beaucoup de temps à lire de la documentation sur internet car je n'avais aucune connaissance en développement web ni dans l'utilisation du module flask. J'ai appris pas mal de concepts à l'aide de tutoriels sur Youtube. J'ai aussi passé plus de temps que je ne l'imaginais à essayer de comprendre mes erreurs en les cherchant sur internet. C'est d'ailleurs, il me semble, une des leçon que j'ai apprises durant ce cours, c'est que la majorité du temps est passée à déboguer du code à l'aide d'internet, et qu'avec les documentation en ligne il est possible de monter assez vite en compétence. Cela reste néanmoins un peu frustrant de passer autant de temps à se documenter et à déboguer pour une fonction (login) qui à l'air en apparence simple à coder.

Même si j'ai l'impression de n'avoir pas avancé autant que je le souhaitais, je suis très content de ce que j'ai appris par l'expérience durant ce projet, je pense vraiment être monté en compétence dans ce domaine, en sachant que je partais avec quelques lacunes.

Isabel Ferreira :

J'étais responsable surtout pour la partie *frontend*, en ce qui concerne le code des pages web et leurs layout, et aussi le lien avec le code en Python. Je me permets de dire que j'ai beaucoup appris pendant ce projet. Je ne connaissais pas Flask et je n'avais jamais utilisé les prompts de mon ordinateur pour faire des installations. J'avais déjà appris à programmer en HTML dans le lycée, mais je ne me souvenais pas très bien, donc j'ai beaucoup pratiqué. Au début c'étais un peu difficile, parce que j'ai dû chercher des tutoriels, essayer plusieurs fois et prendre beaucoup de temps parfois pour trouver des causes des erreurs, mais à la fin je pense qu'on est arrivé à un bon résultat pour le projet. Je pense que les suivis hebdomadaires ont été très importants pour nous aider à maintenir un bon rythme.

Murilo Costa :

J'étais responsable de toute la partie liée à l'acquisition des données financières et des calculs postérieurs pour obtenir notre prix recommandé. Ce projet m'a beaucoup apporté sur comment travailler en groupe sur un projet numérique, via des outils comme github et l' encapsulation du code en différents fichiers. J'ai dû m'approfondir sur des sujets en finance et comment les traiter systématiquement avec python, ce qui j'aimé vu que ça c'est un sujet qui vraiment m'intéresse. J'ai aussi bien aimé apprendre comment utiliser des outils d' extraction de données, vu l'importance que le web scraping a aujourd'hui dans plusieurs entreprises. Finalement, c'était intéressant de travailler avec Flask, vu que j'avais une expérience avec PHP et j'ai pu apprendre un nouveau framework.

Sources

1. JASON FERNANDO, **Discounted Cash Flow.** Disponible à : https://www.investopedia.com/terms/d/dcf.asp#:~:text=Discounted%20cash%20flow%20(DCF)%20is,will%20generate%20in%20the%20future

2. **Using Python, Flask, and Angular to Build Modern Web Apps - Part 1.** Disponible à : https://auth0.com/blog/using-python-flask-and-angular-to-build-modern-apps-part-1/

3. **How To Use Web Forms in a Flask Application.** Disponible à : https://www.digitalocean.com/community/tutorials/how-to-use-web-forms-in-a-flask-application

4. **Bootstrap.** Disponible à : https://getbootstrap.com/docs/5.1/getting-started/introduction/