

Rapport Projet Tuteuré

Z. MELLAH, M. CARDINALE, S. LEFEVRE, A. MIRAS

Année scolaire 2021-2022



Rapport du projet tuteuré de notre année scolaire 2021-2022 au sein de l'IUT de Fontainebleau dans le cadre notre DUT Informatique. Nous y décrivons notre mode d'organisation et les outils que nous avons utilisé. Nous allons présenter notre projet, une application web de recommandation de films.

Table des matières

Introduction	4
Rappel de la nature du projet	4
Problématique	4
Ressources et fonctionnalités du projet	5
Organisation du travail	5
Les outils utilisés	5
Conception: StarUML, JupyterLab	5
Opérations: Trello, Gantt, Scrum, Discord, Git	6
Django	8
Fonctionnalités : rappel cahier des charges	9
Les fonctionnalités implementées et testées	9
Les fonctionnalités abandonnées	10
Structure et description du projet	11
Base de données	11
Répartition du travail	11
Création et connexion au compte	11
Ajout des films dans la bibliothèque	13
L'algorithme de recommandation de film (Zakaria)	14
Processus de développement	14
Changement de sources de données	16
Utilisation de l'API: Cas de l'internaute	17
Utilisation de l'algorithme: Cas de l'utilisateur	17
Résumé en flowchart	18
Difficulté : le temps de calcul	19
Notre choix de l'implémentation minimale	19
Technologies utilisées	19
Conclusions	20
Maxime	20
Compétences technique	20
Compétences humaines	21
Expérience en mode projet	21
Alexis	22
Shana	22
Sources	23

Introduction

Ce document présente et détaille l'élaboration de notre projet tuteuré. Il complète notre cahier des charges rendu cet hivers.

Rappel de la nature du projet

Notre projet est une application web développée avec Django qui permet aux internautes de “gérer” leurs films s’ils sont connectés. Ils peuvent ajouter en favoris, noter, et voir les films recommandés par notre algorithme de recommandation.

L’application est déployée sur Héroku et sur notre base de données sur Dwarves PHP-MyAdmin.

Le plus de notre application est son algorithme de recommandation de films qui sera fonction des œuvres vus, des personnes similaires, des notes et des genres.

Problématique

Nous avons décidé de guider notre rapport à l'aide de la problématique suivante:

En quoi la réalisation d'une application web de recommandation de film peut représenter un projet professionnel concret ?

Bien que l'on soit en apprentissage, peu d'entre nous a eu l'occasion de réaliser des projets informatiques fullstack avec une méthode Scrum. Ce projet est aussi l'occasion pour nous de nous familiariser avec une organisation professionnel d'un projet informatique en utilisant des outils couramment utilisés dans les DSI et les entreprises du milieu informatique.

Ressources et fonctionnalités du projet

Organisation du travail

Afin de parvenir à nos objectifs nous avons décidé de suivre une méthodologie en suivant la méthode Scrum. Zakaria avait le rôle de scrum master, Shana de product Owner, Maxime et Alexis de simple team members. L'objectif était de rester le plus fluide et flexible possible afin de pouvoir aviser notre trajectoire en fonction des difficultés rencontrées. Pour planifier nos tâches nous avons mis en place un trello. Cela nous a permis d'avoir une vision claire sur les tâches effectuées et celles à venir. Nous échangions à l'écrit et organisions nos réunions vocales grâce à un serveur discord créé pour le projet. Nous mettions également notre travail en commun grâce à un [repo git](#).

Nous avions initialement défini un planning à respecter afin de lisser la charge de travail tout au long de l'année. Afin de suivre l'évolution de notre projet, nous avions fait un diagramme de Gantt (*cf : cahier des charges*). Malheureusement, nous n'avons pas réussi à respecter les délais fixés. En effet nous avons eu du mal à travailler sur ce projet en plus des autres travaux à effectuer en parallèle. Le rythme de l'alternance a également joué un certain rôle important dans notre désorganisation en nous fatiguant et nous coupant totalement des cours. Nous avons souvent eu du mal au cours de l'année à nous fixer des dates de fin de sprint et à respecter ces deadlines. De plus, à cause de nos emplois du temps respectifs nous avons souvent eu du mal à organiser des meetings ou à nous retrouver tous ensemble dans ces réunions.

Les outils utilisés

Conception: StarUML, JupyterLab

StarUML

Pour réaliser les diagrammes de classe et de séquence, pour l'algorithme, entre autres, nous avons utilisé StarUML, un outil que nous utilisions déjà en cours d'ACDA avec M. Hernandez. Entre le cahier des charges et l'élaboration du projet, nous avons du modifier la base de données (*cf : Base de données*).

JupyterLab

JupyterLab a été utilisé par Zakaria lors de la phase de conception de l'algorithme. Nous détaillons cette utilisation plus en détail dans la partie Processus de développement.

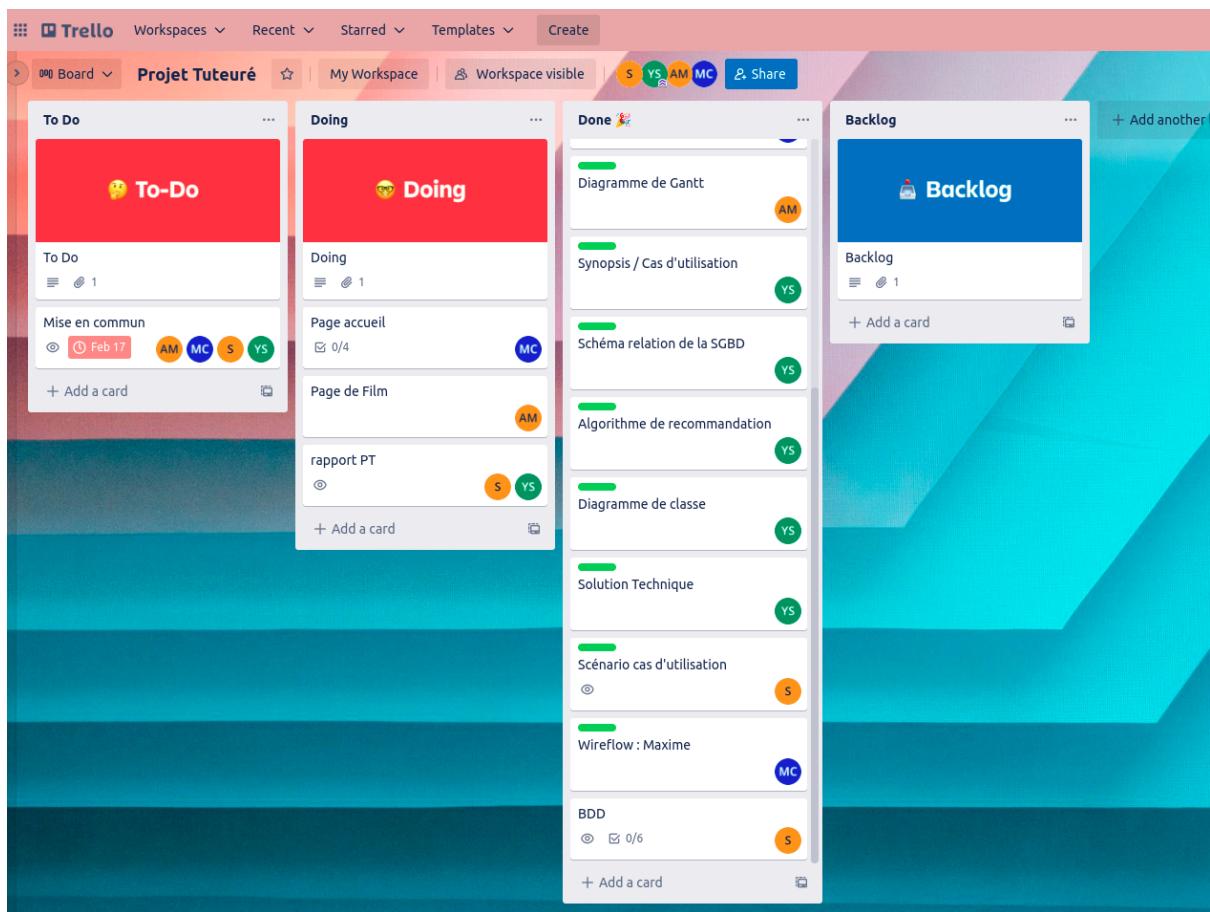
Opérations: Trello, Gantt, Scrum, Discord, Git

Gantt

Afin de suivre l'évolution de notre projet, nous avons réalisé un **diagramme de Gantt** (*cf : cahier des charges*). Malheureusement, nous n'avons pas réussi à respecter les délais fixés. Nous avons donc replanifier le diagramme de Gantt sur un délai “plus court”.

Trello

Pour la répartition du travail, nous avons utilisé et mis à jour le **Trello** au fur et à mesure que nos tâches évoluaient.

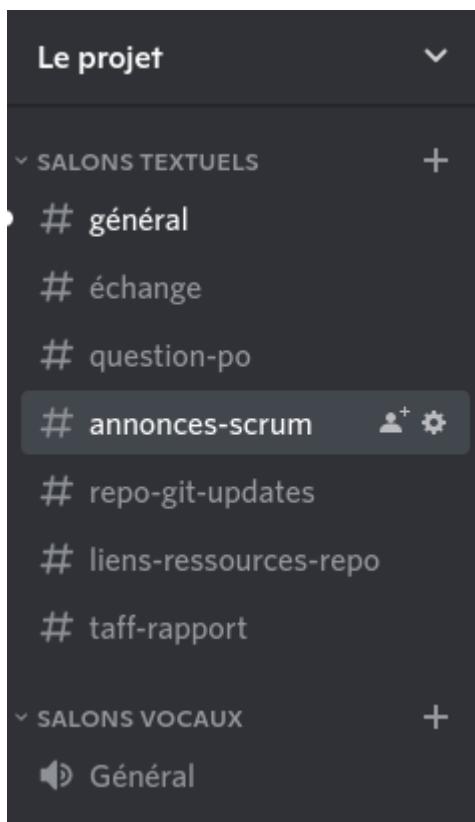


Nous utilisions un système inspiré de la méthode **Kanban** avec 4 sections:

- ToDo: ce qu'il reste à réaliser
- Doing: ce qui est en cours de réalisation
- Done: Ce qui est réalisé
- Backlog: Ce qui est en file d'attente

Ce **Trello** nous a beaucoup servi au début du projet, notamment pour bien fixer notre partage des tâches. Toutefois, nous l'avons progressivement délaissé en faveur de discussion et de message Discord épingle. Comme nous étions plus souvent sur Discord cela nous paraissait plus simple de favoriser un mode d'organisation basé sur cet outil.

Discord



Pour rester en contact, nous avons mis en place un serveur Discord où nous nous réunissions régulièrement, à la fois en chat textuel et vocal. C'est sur ce serveur que le Scrum Master annonçait les prochains rendez-vous et organisait le planning.

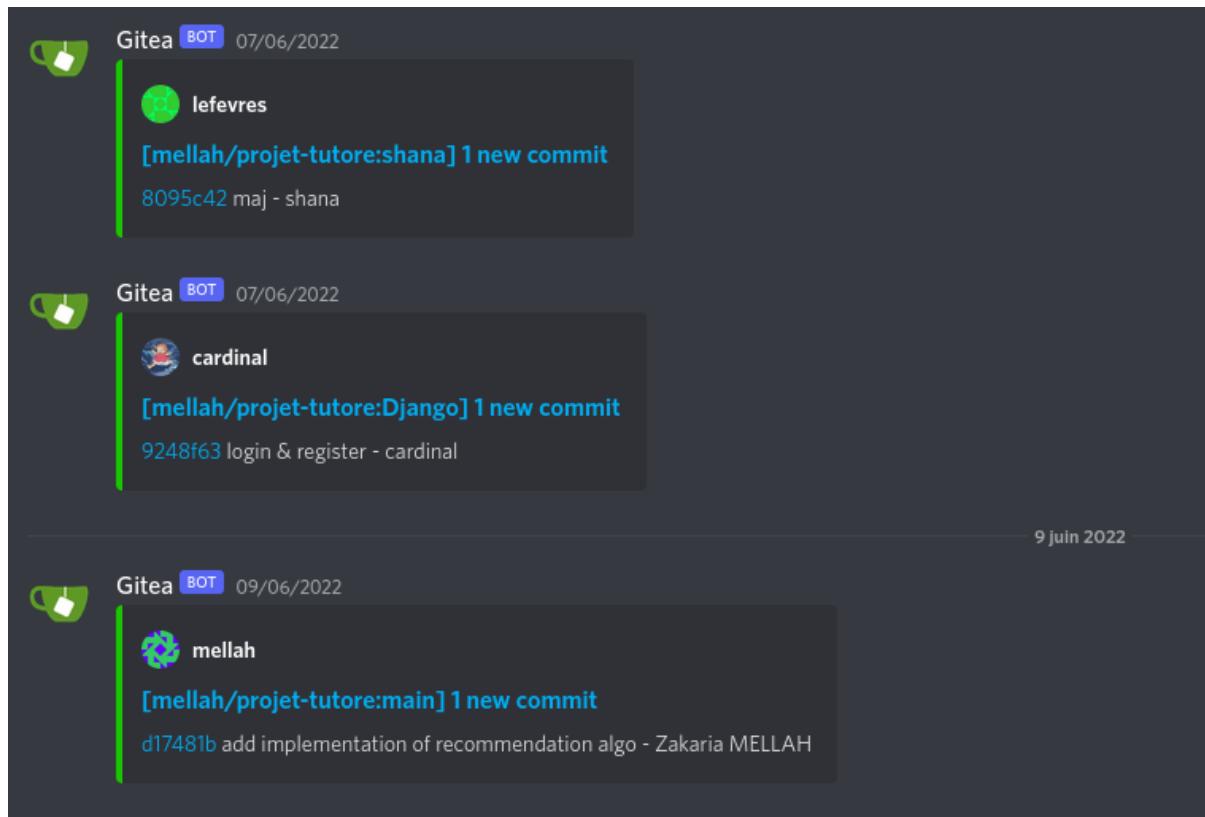


Scrum

Durant toute l'année, nous avons utilisé la **méthode agile scrum**. Sur la dernière période de cours (vers juin), nous avons rapproché les sprints jusqu'à 2-3 jours. Durant l'année, nous étions sur une fréquence de sprints d'un mois.

Git

Afin de travailler convenablement de façon individuelle tout en facilitant la collaboration et la mise en commun du code nous avons bien sûr utilisé le système de gestion de version distribuée **Git**. Nous l'avions déjà utilisé pour d'autres projets à durant notre scolarité. Il était donc relativement simple à prendre en main. Pour le dépôt distant, nous avons utilisé le serveur de l'IUT qui contient également une interface dédiée via **Gitea**. Il est disponible à cette adresse : <https://dwarves.iut-fbleau.fr/gitiut/mallah/projet-tutore>. La **gestion** du dépôt distant Git a été laissé à Zakaria. Il s'occupait de faire les merge et de gérer les éventuels conflits. Il a également mis en place un système de **protection de branche** pour éviter des erreurs d'inadveritance. Nous créons des branches selon les "besoins" du projet. *Ex: Cahier des charges, main, Django, shana (pour le rapport, requête base de données), etc.* . Enfin, nous avons également utilisé un **webhook Discord** nous permettant de poster une notification sur un channel Discord lors d'un push sur le serveur distant.



Django

Pour cette application web nous avons fait le choix d'utiliser le framework python open source Django. Le but est évidemment de rendre le développement simple et rapide. Ce framework se base sur le modèle MVC, il prodigue une solution de remapping d'URL et offre génère automatiquement une API d'accès aux données qui permet de ne pas écrire des requêtes SQL associées à des formulaires puisqu'elles sont générées automatiquement. En plus de cela, le framework prends totalement en compte la sécurité de l'application en se protégeant par exemple des injections SQL ou des clickjacking et en utilisant entre

autres le header Host pour construire des URLs **Django** permet aussi de se connecter simplement aux bases de données réalisées afin d'afficher les informations nécessaires à notre site web. Il a fallu choisir un pilote qui implémente l'API python de base de données, le choix c'est vite orienté vers **mysqlclient** car il est plus complet pour ce que l'on souhaite faire. Ces pilotes respectent la concurrence entre fils d'exécution et gèrent le regroupement de connexions. Une fois la liaison effectuée nous pouvions utiliser ces données. ### Base de données

Nous hébergeons notre applications sur **Heroku**.

Toutes nos tables sont sur **PHPMyAdmin** de Shana sur le serveur DWarves. Ces dernières commencent par “**PT_<table>**” car d'autres tables d'autres projets sont présentes dans la base et cela permet de structurer les différentes données.

	id_like	id_movie	id_member
1	5	40	1
2	6	19	4
3	7	242	5
4	8	344	1

Pour réaliser des tests sur les fonctionnalités du site web et de la base de données, nous avons hébergé la base de données avec la pile **XAMPP**.

Fonctionnalités : rappel cahier des charges

Les fonctionnalités implementées et testées

Ci-dessous un tableau représentant les tests effectués sur notre application. Nous confrontons ce qui devait être prévu, indiqué dans le cahier des charges et notre application “finale”.

Prévu par le cahier des charges	test réussit	Raison
l'internaute cherche un film		
l'internaute peut voir les films qui lui sont recommandés		
l'internaute peut accéder aux différentes informations des personnes		
l'internaute peut accéder aux différentes informations des films		
l'internaute veut s'inscrire	OUI	
l'internaute veut se connecter	OUI	
l'internaute se déconnecte		

Prévu par le cahier des charges	test réussit	Raison
utilisateur connecté ajoute un film dans ses favoris et ses vus recommandation d'un film selon les films aimés et vus		

Les fonctionnalités abandonnées

Prévu par le cahier des charges	test réussit	Raison
recommandation d'un film par rapprochement d'utilisateur	NON	Délai court
déploiement home-serveur de Zakaria	NON	Délai court

Structure et description du projet

Base de données

Comme indiqué précédemment, toutes nos tables commencent par “PT_”. Après concertation et réévaluation des objectifs pour le projet, nous avons décidé de modifier le schéma de la base de données initialement prévue.

Répartition du travail

Zakaria a réalisé de l’algorithme de recommandation de films. Maxime et Alexis ont créé les vues de l’application. Shana s’est occupé de toute la partie base de données et de la majeure partie du rapport. Les parties techniques du rapport ont été rédigé par les personnes respectives afin de faciliter leur rédaction.

Création et connexion au compte

Les internautes sont tout d’abord accueilli par cette page. Ils peuvent choisir de s’inscrire, de se connecter ou bien, juste de “naviguer” sur l’application.

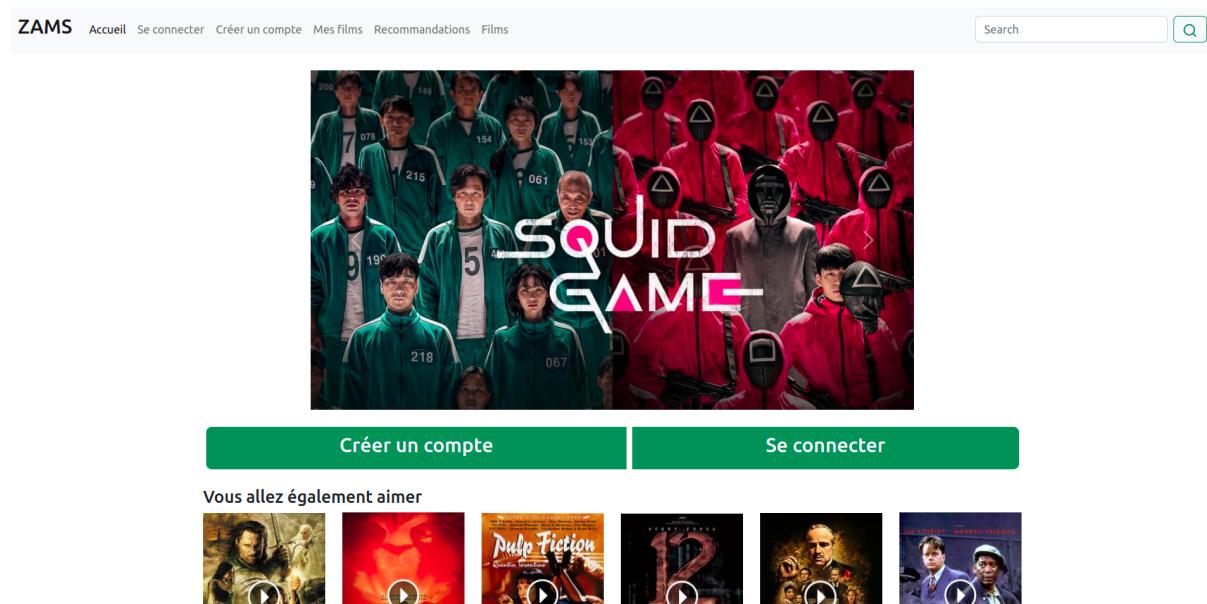


Figure 1: image accueil internaute

L'inscription permet profiter des avantages que propose notre application : recommandation de films, ajout dans une bibliothèque de favoris, vus. Tous les membres sont dans la table **PT_Member**. Une simple requête **insert** nous permet de mettre à jour cette table. L'email et le pseudo sont uniques. Chaque tuple possède un clé afin de pouvoir mettre en corrélation les autres tables. Pour la connexion, on parcourt la table et on vérifie qu'il existe et que le mot de passe correspond au pseudo/email entré.

Ci-dessous, la page où les internautes peuvent s'inscrire après avoir cliqué sur "Se connecter" sur le bouton principal ou sur la barre de recherche.

Adresse email

Nom d'utilisateur

Mot de passe

J'ai lu les CGU

Se connecter

Voici la page où les internautes peuvent se connecter. Respectivement, la même méthode que pour s'inscrire.

Adresse email / Nom d'utilisateur

Mot de passe

Se souvenir de moi

Se connecter

Ajout des films dans la bibliothèque

Les tables **PT_Like** et **PT_View** sont associatives. Elles prennent respectivement l'id du film et l'id de l'utilisateur. Nous utilisons les **vues relationnelles** que nous propose PHPMyAdmin afin de mettre en corrélations nos données, savoir “qui appartient à qui”.

Il faut que l'utilisateur soit connecté pour accéder à cette page :

ZAMS Accueil Se connecter Crée un compte Mes films Recommandations Films Search

Open this select menu ▾

Catégorie 1

 Titre Titre Titre Titre Titre Titre Titre

Catégorie 2

 Titre Titre Titre Titre Titre Titre Titre

Catégorie 3

 Titre Titre Titre Titre Titre Titre Titre

Voici comment s'organise notre base de données :

+ Options		id_movie	title	genres
<input type="checkbox"/>	Éditer Copier Supprimer	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
<input type="checkbox"/>	Éditer Copier Supprimer	2	Jumanji (1995)	Adventure Children Fantasy
<input type="checkbox"/>	Éditer Copier Supprimer	3	Grumpier Old Men (1995)	Comedy Romance
<input type="checkbox"/>	Éditer Copier Supprimer	4	Waiting to Exhale (1995)	Comedy Drama Romance
<input type="checkbox"/>	Éditer Copier Supprimer	5	Father of the Bride Part II (1995)	Comedy
<input type="checkbox"/>	Éditer Copier Supprimer	6	Heat (1995)	Action Crime Thriller
<input type="checkbox"/>	Éditer Copier Supprimer	7	Sabrina (1995)	Comedy Romance
<input type="checkbox"/>	Éditer Copier Supprimer	8	Tom and Huck (1995)	Adventure Children
<input type="checkbox"/>	Éditer Copier Supprimer	9	Sudden Death (1995)	Action
<input type="checkbox"/>	Éditer Copier Supprimer	10	GoldenEye (1995)	Action Adventure Thriller
<input type="checkbox"/>	Éditer Copier Supprimer	11	American President, The (1995)	Comedy Drama Romance
<input type="checkbox"/>	Éditer Copier Supprimer	12	Dracula: Dead and Loving It (1995)	Comedy Horror

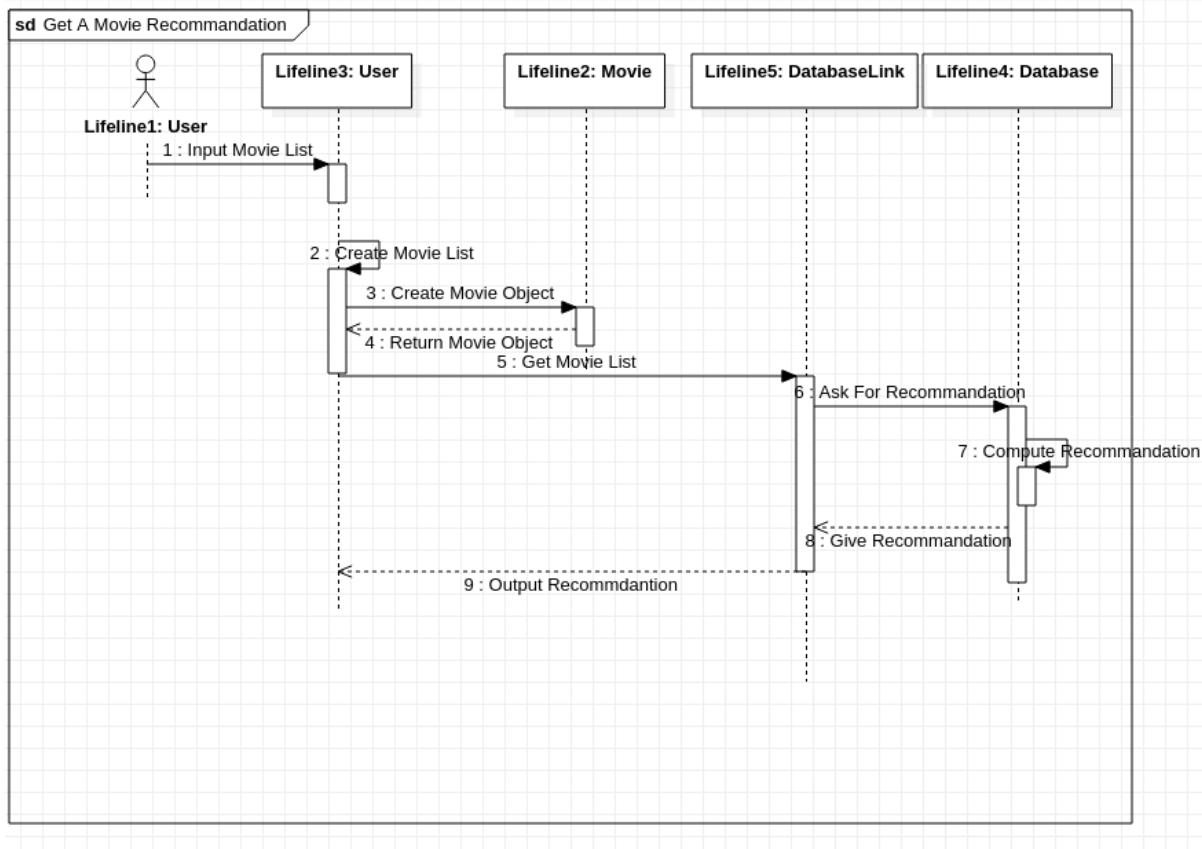
L'algorithme de recommandation de film (Zakaria)

L'algorithme de recommandation est une des fonctionnalité centrale de notre application. Comme décrit dans le cahier des charges, notre algorithme se repose sur des données différentes en fonction du profil de l'utilisateur:

- Un internaute (non connecté) se vera proposer une sélection de films **populaires actuellement**
- Tandis qu'un utilisateur connecté se vera proposer un film en fonction des films qu'il a **vu et aimé**

Processus de développement

Une fois la stratégie globale et les objectifs fixés, Zakaria s'est attelé à la conception et l'implémentation de l'algorithme. Tout d'abord, il a décrit ses fonctionnalités puis écrit un **diagramme de séquence** pour le cas où un *utilisateur connecté* demande une recommandation:



Ensuite, Zakaria a beaucoup utilisé **JupyterLab** afin de compléter un **PoC (Proof Of Concept)** rapidement. Cet outil lui a aussi permis de réduire le cycle de développement: Écriture -> Test -> Correction.

```

[7]: import pandas as pd
# Get user Fav movies list
user_fav_movies = pd.read_csv('src/fav-movies.csv')

# Get movie db
movie_db = pd.read_csv('ml-latest/movies.csv')

# Only get movies seen by the user
fav_movies_info = pd.merge(left=user_fav_movies, right=movie_db, on='movieId')

# Get genres
fav_genres = fav_movies_info['genres']

# Get most popular genre
genres_dict = {}
for row in fav_genres:
    row = row.replace('|', '')
    for genre in row:
        if genre in genres_dict:
            genres_dict[genre] += 1
        else:
            genres_dict[genre] = 1

max_value = 0
most_popular_genre = ''
for genre in genres_dict:
    if genres_dict[genre] > max_value:
        max_value = genres_dict[genre]
        most_popular_genre = genre

[27]: # Get Movie of same genre
most_popular_genre_movies = pd.DataFrame(columns=movie_db.columns)
for index in range(len(movie_db)):
    if movie_db['genres'].str.contains('Animation') != 1:
        # most_popular_genre_movies = most_popular_genre_movies.append(movie_db.iloc[index])
        most_popular_genre_movies = pd.concat([most_popular_genre_movies, movie_db.iloc[index].to_frame()], ignore_index=True)
        break

print(most_popular_genre_movies)
      movieId          title           genres
0       1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy

[165]: movie_ratings=pd.read_csv('ml-latest/ratings.csv')

# Get average ratings of theses movies
most_popular_genre_movies_ratings = pd.merge(left=movie_ratings, right=most_popular_genre_movies, on='movieId')

avg_ratings = most_popular_genre_movies_ratings.groupby(['movieId','title'])['rating'].mean()

# get most popular movie
recommended_movie = avg_ratings['rating'].idxmax()

# get rating
recommended_movie_rating = avg_ratings.loc[avg_ratings['rating'].idxmax()]

print("We recommend you " + recommended_movie['title'] + " Which has an average rating of " + str(recommended_movie_rating))

We recommend you : Lotte and the Moonstone Secret (2011), Which has an average rating of : 5.0

```

L'organisation en **cellules indépendantes** de JupyterLab permet une **programmation modulaire et très flexible** qui lui a permis d'arriver à un résultat satisfaisant sans trop d'effort.

À l'issue de ces deux étapes, Zakaria a pu terminé un premier jet de l'algorithme qui sera

une base pour celui qui sera effectivement implémenté sur le site.

Changement de sources de données

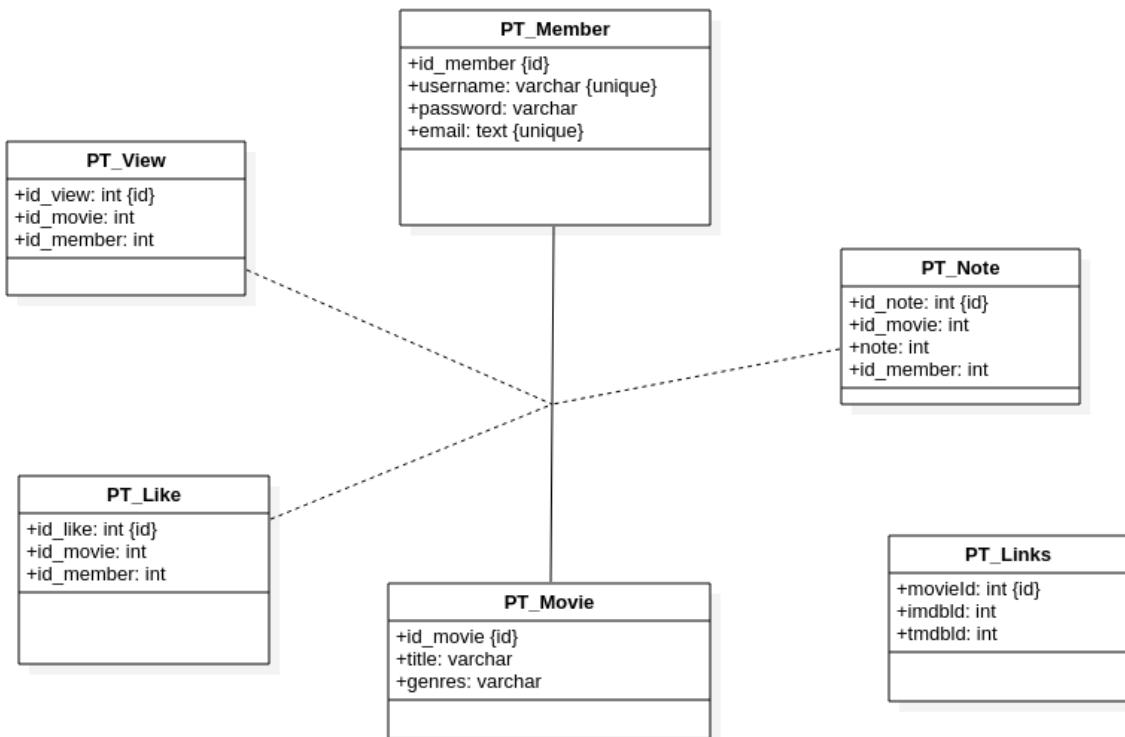
Une fois l'algorithme terminé, nous avions prévus de commencer le lien entre la base de données de **MovieLens** avec celle d'**IMDB** afin de récupérer les données manquantes dans l'un mais présente dans l'autre.

Le lien est assez simple à réaliser. En effet, les données de MovieLens contiennent une table nommé **links** où chaque movieId de la base de MovieLens est associé à son équivalent dans **IMDB**.

	A	B	C
1	movielId	imdbId	tmdbId
2	1	114709	862
3	2	113497	8844
4	3	113228	15602

C'était ce que nous comptions utiliser au début, c'est à dire récupérer le cast et toutes les données liées (date, langue, genre) à un film.

Toutefois, nous avons découvert l'API de **TMDB** durant un projet de WIM avec M. Monnerat. Cet API nous permet de récupérer toutes les données liées à un film mais il nous faut seulement intégrer le tableau **links** à notre schéma de base de données.



Une fois cela fait, nous avons pu recalibrer notre projet. Pour certaines parties, nous utiliserons l'API de TMDB(Internaute) et pour d'autres nous utiliserons notre algorithme (Utilisateur).

Utilisation de l'API: Cas de l'internaute

Pour l'internaute, les films en entrée sont simplement les films les plus populaires choisies via l'API de **TMDB**.

Get Popular

GET /movie/popular

Get a list of the current popular movies on TMDB. This list updates daily.

Utilisation de l'algorithme: Cas de l'utilisateur

Pour l'utilisateur connecté, la recommandation se fait grâce à notre algorithme en utilisant la base de données de **MovieLens**. L'utilisateur doit avoir une liste de films préférés, cette liste est récupérée via l'id de l'utilisateur dans notre base de données. Une fois ceci fait, le serveur charge les genres des films préférés puis sélectionne le genre le plus populaire parmi ceux-là (il réalise un simple décompte). Ensuite, on sélectionne des films du même genre ainsi que leur note. Enfin, on sélectionne parmi cette dernière liste le film le plus populaire.

Le résultat obtenu n'est que le nom et l'id du film. Il faut par la suite obtenir les informations additionnelles (cast, date,...). Pour cela, on utilise une nouvelle fois l'API de **TMDB** en utilisant la requête:

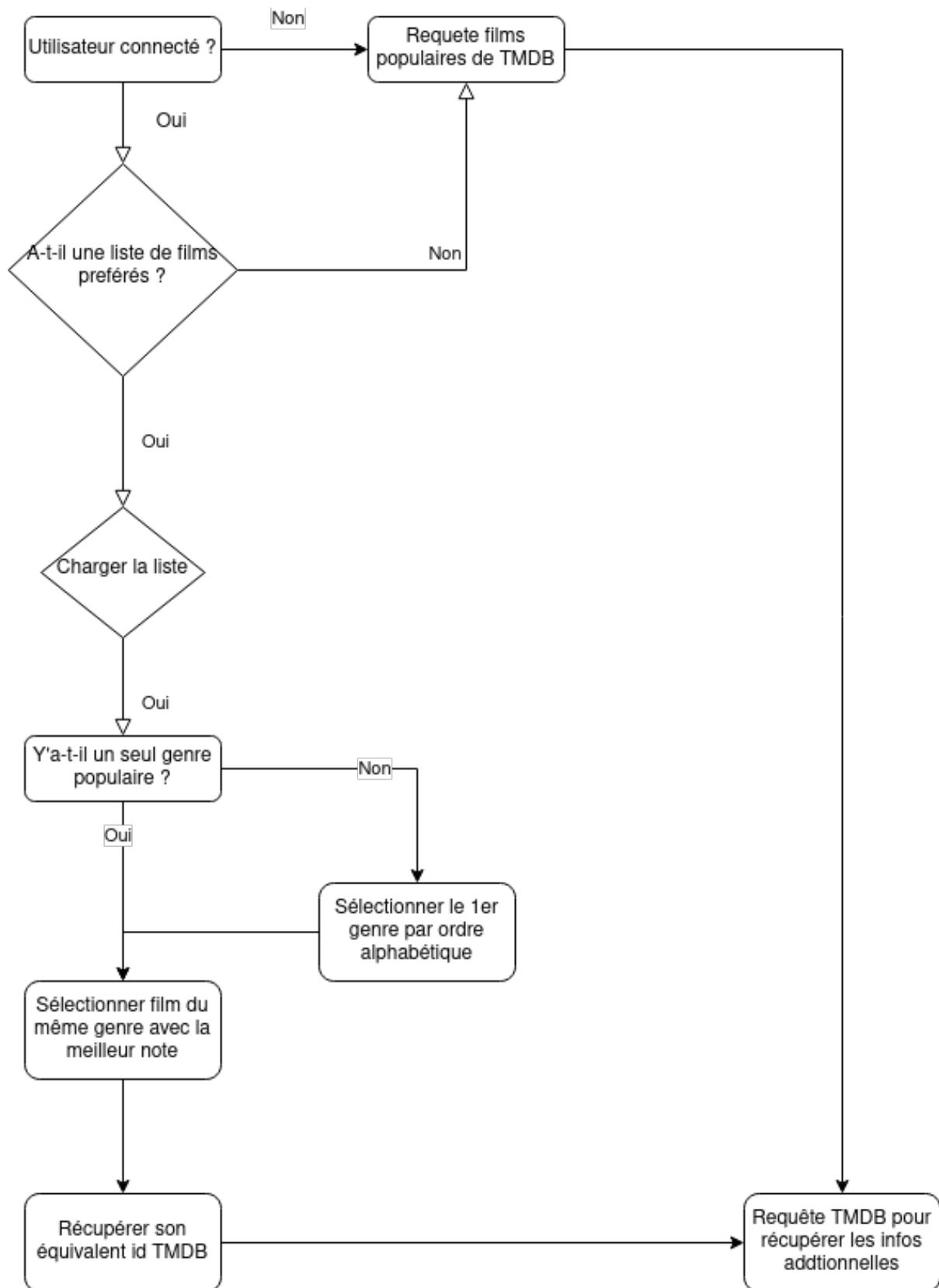
Get Credits

GET /movie/{movie_id}/credits

Get the cast and crew for a movie.

Qui nous permet de récupérer entre autres le réalisateur, la langue originale du film etc,...

Résumé en flowchart



Difficulté : le temps de calcul

Un des challenges de l'implémentation de l'algorithme fût la vitesse d'exécution et notamment la durée de lecture des requêtes qui pouvaient être très important lors des tests. En effet, Zakaria a testé l'algorithme en local en lisant les fichiers **csv** (source des tableaux de la base de données), ce qui était très pénalisant au niveau du temps de calcul. Il a pensé utiliser un calcul multi-threadé car il craignait que le problème se pose en production. Heureusement, l'utilisation d'une BD dédiée a réglé le problème sans nécessiter de solution particulière.

Notre choix de l'implémentation minimale

Il est possible de lier les deux algorithmes présentés dans le cahier des charges (en fonction du contenu et du rapprochement d'utilisateur) mais nous avons décidé d'implémenter uniquement la recommandation en fonction du **contenu** c'est-à-dire l'implémentation minimale. En effet, par manque de temps, nous n'avons pas réalisé l'implémentation idéale car l'aspect réseau social du site n'était pas encore assez mature.

Technologies utilisées

Nous avons utilisés la librairie **pandas** qui est incontournable lors de la manipulation de données. Lors des tests, Zakaria a pu utiliser la librairie **richText** afin d'avoir une interface agréable. Nous n'avons pas pu utiliser la méthode de factorisation des matrices Utilisateurs * Films de la librairie **Surprise** car l'implémentation minimale a été retenu.

Conclusions

Comme dit précédemment nous avons eu du mal à nous organiser comme nous aurions dû. Pour palier au retard accumulé nous avons décidé de revoir nos objectifs à la baisse. Après une analyse des capacités prévues, nous en avons convenu qu'il était impossible de ne pas implémenter l'algorithme de recommandation que l'on considérait comme le cœur de notre application. En revanche nous nous sommes accordés sur la mise de côté de l'aspect réseau social du site web. Nous avons convenus que c'était la partie qui demanderait le plus de temps, ressource dont nous disposions qu'en quantité très limitée. Hormis cela nous sommes parvenus à implémenter tout le reste. Pour ce qui est de l'algorithme il aurait toujours été possible d'en faire un meilleur mais nous avons réussi à faire quelque chose qui d'assez poussé et en même temps pas trop demandant en temps de développement pour ce projet.

Maxime

Au cours de ce projet j'ai régulièrement été frustré d'apprendre, un peu tard, des choses qui auraient pu me servir à concrétiser ce projet. L'exemple le plus récent est la découverte des APIs de données en cours de WIM. Si nous avions connaissance de ces APIs au commencement du projet les aurions probablement utilisé, et pensé notre application totalement différemment. . De plus, la fragmentation du rythme de travail imposé par l'alternance pour un projet si long fût un système absolument néfaste pour ma motivation à travailler sur ce sujet.

Toutefois, comme pour toute expérience il faut savoir en tirer de bonnes choses. Alors même si ce projet n'était pas le plus agréable à réaliser. Il m'aura mieux appris à utiliser la méthode Scrum, il m'aura fait découvrir Django et m'auras permis d'appliquer un grand nombre de choses appris en cours comme la création de wireflows, l'utilisation d'outils comme trello, ou tout simplement la programmation web. Comme d'autres projets auparavant il m'aura de nouveau souligner l'importance de l'organisation et du respect du planning. Je pense toutefois que nous avions une bonne organisation, mais que nous n'avons pas réussi à l'adapter / nous adapter nous même au cadre de travail et son rythme particulier. ## Zakaria

Compétences technique

En ce qui me concerne, ce projet fût l'occasion de découvrir Python plus en profondeur et le framework Django que je n'avais jamais utilisé. J'ai pu me documenter sur les algorithmes de recommandation, avec les outils à ma disposition pour les implémenter bien que je n'ai pas pu les utiliser. (librairie Surprise) J'ai beaucoup apprécié la découverte de

JupyterLab, ça m'a grandement aidé durant la conception de l'algorithme et les multiples tests que ça implique. J'ai donc pu améliorer mon autonomie sur de nombreux sujets et apprendre à me documenter par moi-même.

Côté serveur, j'ai apprécié l'idée de déployer l'application sur mon serveur personnel. Toutefois, cela demandait beaucoup d'entretien technique et de temps, j'ai pris la décision de déployer sur un fournisseur cloud pour m'éviter de mauvaises surprises et faciliter la livraison des différentes versions du projet. J'ai quand même pu me frotter aux diverses manipulations impliquant le déploiement d'une application et sa mise à disposition sur Internet. (ouverture du pare-feu, configuration du serveur,...)

Compétences humaines

Durant ce projet, j'ai pu améliorer mes compétences de communication. En tant que Scrum Master, j'avais pour responsabilité d'organiser et d'animer les cérémonies agile. Durant nos rendez-vous, il me fallait faire en sorte que chacun ait une occasion de donner son avis et d'être moteur du projet.

Je devais faire en sorte que les horaires de rendez-vous soient connus de tous et m'assurer que chacun soit disponible. Surtout en période de fin de projet où la cadence s'accélère très rapidement.

Lors de difficultés il faut savoir rester calme et tranquillement se remettre au travail en étant concentré sur l'objectif à poursuivre. Il peut s'avérer trop facile de pointer les responsabilités de chacun mais il ne faut oublier le projet pour ne pas dévier.

Expérience en mode projet

Il a fallu composer avec l'alternance, ce qui ne fut pas facile. En effet, le découpage des séquences de cours et d'entreprise rendait l'organisation plus difficile et surtout pouvait ralentir le rythme. Nous avons tous plus ou moins reconnus que l'organisation est primordiale dans un projet et encore plus dans un projet informatique, qui peut être d'une grande complexité technique.

Il fallait aussi s'adapter à trois modes de travail: - En cours - En entreprise - En PT

Parfois, jongler entre ces trois était compliqué. Avec la grande amplitude que permet un tel projet il est facile de se retrouver perdu par la masse de travail à réaliser. Surtout, le plus compliqué a été pour moi de garder une vue globale du projet.

Toutefois, ce projet fut une bonne expérience et m'a permis de me familiariser avec les outils utilisés en entreprise (ce qui n'est pas mon cas dans mon entreprise) ainsi qu'avec la méthode agile mais sur le temps long.

Alexis

Ce projet a été très différent de tout ce que j'ai pu réaliser jusqu'à aujourd'hui. Le contexte particulier dû à l'alternance a rendu la tâche un peu plus difficile en matière d'organisation, la durée de projet très longue pouvait rendre cela sournois. Cela nous a démontré que l'organisation est primordiale dans un tel projet. Ce que je retiens c'est que pour réaliser un tel projet la technique ne suffit pas il faut avoir une bonne gestion globale du groupe. Les difficultés surmontées m'ont apporté de l'expérience et de nombreuses connaissances technique et de gestion, de plus l'application de compétence acquise durant le DUT et en entreprise permet de mieux se rendre compte de nos compétences actuelles.

Shana

Pour ma part, c'est un projet très intéressant car il rassemble beaucoup d'aspects : longue durée, élaboration d'un cahier des charges, travail à plus de deux, réalisation de A à Z d'un projet, etc. En effet, réaliser un projet à deux, comme l'année dernière, n'est pas du tout pareil qu'à quatre. Il faut beaucoup plus d'organisation. De plus, le projet est sur une longue période, il faut resté motivé, assidus, à l'écoute tout au long de l'année, ce n'est pas évident avec la formation, les autres projets et la vie personnelle. Cependant, nous avons su nous rendre disponibles malgré nos emplois du temps bien différents. J'ai pu découvrir un petit peu Django. Nous avons décidé de répartir le travail en fonction de nos points forts. Dans ce genre de travail, chaque rôle est important. Les collègues comptent sur nous et "dépendent" de nous pour le rendu. Ce n'est pas un travail personnel, nous avons du "instaurer" une confiance de travail. Je pense que nous avons rempli cette "part de marché".

Pour conclure, toute expérience est bonne à prendre afin de progresser, que ce soit au niveau du développement et de l'organisation du travail. Si j'avais à refaire ce genre de projet, j'améliorai ma communication. Nous avons eu quelques difficultés au début du projet à communiquer. Dès le projet concrétisé, la situation fut meilleure.

Sources

[StackOverflow](#)

[AskCodez](#)

[Docs Django](#)