Projet SI

PolyWatch

BURNIER-FRAMBORET Alexandre

CHARLET Thomas

CHEVALLIER Guillaume

DE LA FUENTE Martin

HILLERITEAU Rémi

MAURICE Valentin

Table des matières

1) Contexte de la réalisation

1.1) Contexte

C'est un fait selon une étude datant de 2017, les français sont « accros » aux media et plus particulièrement à la télévision. Cette étude montre qu'en moyenne un français passe un peu moins de 4 heures à regarder la télévision par jours. De plus, le sondage effectué permet de faire ressortir que 94% des foyers français possédaient une TV et approximativement 56% de cette même population avait à sa disposition une TV connectée. Par ailleurs, les tendances actuelles montrent que ce pourcentage de TV connecté est amené a augmenté puisque en 2018, un peu plus de 60% des foyers français possédaient ce type de produit.

Ces chiffres nous permettent à ce jour d'établir l'importance des guides media TV sur Internet. Ces derniers agissant comme catalogue permettant à un utilisateur de visualiser les programmes proposés sur la TV accompagnés de services de replay ou encore de streaming. De nombreux guides TV existent actuellement sur Internet et permettent aux utilisateurs de disposer des informations précédemment évoquées comme « Télé Loisir ». Avec la montée des plateformes VOD et streaming de nouveaux catalogues apparaissent sur Internet tel que « Netflix » ou encore « Amazone Prime Vidéo» permettant cette fois-ci aux utilisateurs de regarder leurs programmes préférés en différé.

Une problématique à explorer serait donc la mise en commun des informations provenant de ces deux types de catalogues représentant les flux linéaires et non-linéaires (TV/VOD) mise à disposition des utilisateurs.

Pour cela nous réaliserons un portail web chargé de mixer ces informations (TV/VOD) à destination des utilisateurs. Néanmoins la création d'un tel projet nous confrontent à divers enjeux et problématiques sur lesquels nous reviendrons dans ce rapport.

Le portail Web créée sera disponible sur différentes plateformes (ordinateur, smartphone) par l'intermédiaire d'un explorateur Web et devra être en capacité de gérer un grand nombre d'utilisateurs. La mise à jour du contenu de la plateforme devra s'effectuer régulièrement étant donné le caractère « changeant » du programme TV au fil de la journée. L'idée étant de proposer les informations les plus « pertinentes » aux utilisateurs de la plateforme, critère sur lequel nous reviendrons par la suite.

L'architecture sur laquelle se fonde notre application Web doit donc permettre de répondre à l'ensemble de ces enjeux en garantissant des performances semblables sinon proche des catalogues TV déjà existant.

1.2) Objectifs

L'objectif de ce projet est donc de proposer une plateforme Web agissant comme guide média regroupant des informations provenant des flux linéaires (TV) et non-linéaires (VOD). La plateforme se présentera sous la forme d'un portail média garantissant un niveau de performance proche des guides TV déjà existant sur Internet. Nous avons choisi dans le cadre de ce projet de prendre les informations du catalogue VOD NETFLIX comme exemple de flux non linéaire à traiter.

Cette plateforme permettra notamment de visualiser les informations des deux types de flux mixées et présentées selon leur « pertinence », critère notamment basé sur l'heure de diffusion du programme TV et la nouveauté des programmes VOD. Des filtres seront également proposés à l'utilisateur pour lui permettre d'affiner sa recherche.

1.3) Hypothèses

Une première étude consistant à établir l'architecture du projet ainsi que les technologies à utiliser pour créer notre application est à mener afin d'être en mesure d'assurer le critère de performance fixé comme contrainte. Nous avons donc choisi de mettre en place une base de données permettant de stocker les informations des fichiers XMLTV et données VOD afin d'être en mesure de proposer un système évolutif.

1.4) Bases Méthodologiques

Notre projet a été réalisé dans la continuité du travail fourni par des équipes des années précédentes ayant pour objectif de créer un catalogue ne comportant que les flux linéaires (TV). Nous nous sommes donc inspirés de ces projets passés notamment pour la partie récupération et stockage des flux linéaires.

Pour cela nous avons adopté le standard XMLTV pour tout ce qui concerne la récupération des informations propres au programmes TV stockés au format XML. Le choix de respecter cette représentation des données est nécessaire voir indispensable à la création d'un modèle de stockage de l'information non chronophage, du moins pour la partie flux linéaire. Il est maintenant nécessaire d'ajouter à ce modèle l'aspect flux non linéaire (VOD). Nous avons donc compléter le modèle de données fournis dans ce sens.

Ce projet a été réalisé en suivant une méthode agile, nous permettant d'échanger sur l'évolution du projet au cours du temps avec Mr. DELALANDRE, MOA de notre projet. Ces retours ayant pour but de nous conforter des les choix adoptés et communiquer les avancées du projet.

Compte tenu des problématiques et enjeux précédemment évoqués, il était nécessaire d'effectuer dans un premier temps une phase d'études des technologies nous permettant de créer notre application. Dans cette optique nous avons orienté nos recherches vers des

technologies Front-End nous permettant de proposer une visualisation fluide et efficace des données ainsi qu'un système de stockage permettant d'établir des règles de tries et filtres sur les programmes stockés. Le choix d'une base de données relationnel nous a été suggéré par notre encadrant. Par ailleurs, aucun langage de programmation n'a été imposé nous laissant ainsi le choix de ce dernier.

Équipe de développement et répartition des tâches :

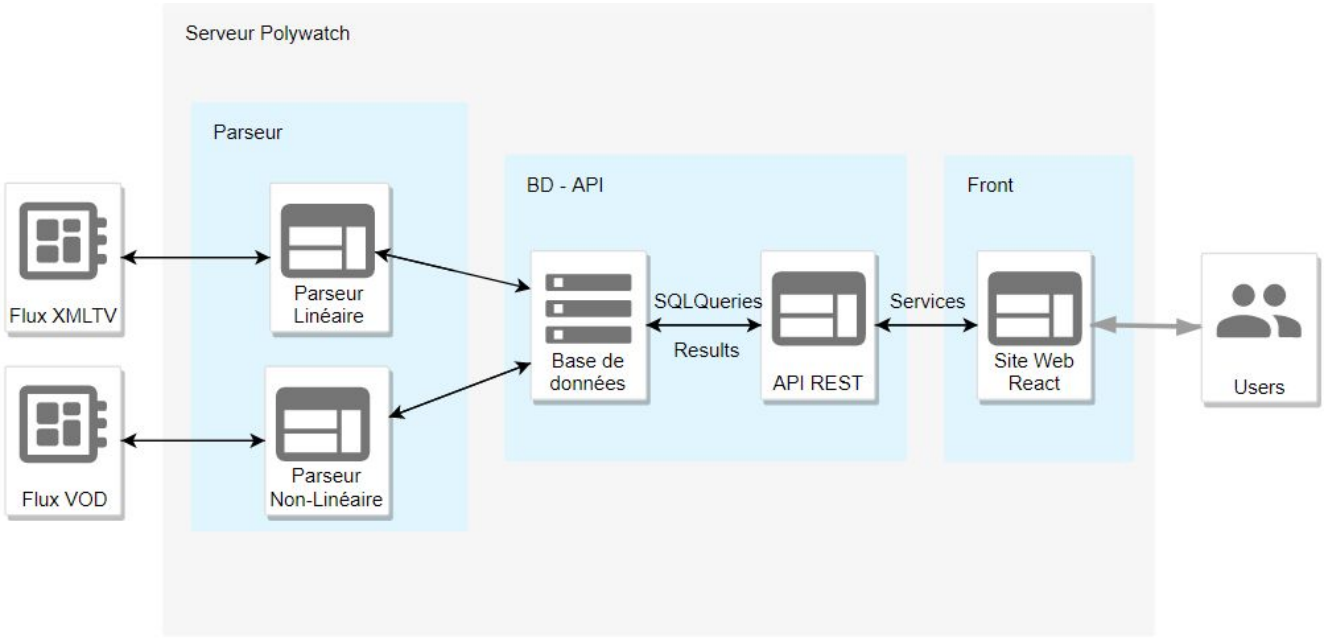
- Parseur TV/VOD et Injection des données dans la base de données :
    - Burnier-Framboret Alexandre,
    - Chevallier Guillaume,
- Création, administration base de données relationnel (MySQL), création API Rest :
    - Charlet Thomas,
    - De la Fuente Martin,
- Implémentation des vues utilisateurs (Framework ReactJs)
    - Hilleriteau Rémi,
    - Maurice Valentin,

2) Description générale

2.1) Environnement du projet

Comme nous avons pu l'évoquer précédemment, l'objectif est de permettre d'accéder à notre portail PolyWatch quel que soit le terminal utilisé. Notre application étant un site Web, sa compatibilité est indépendante du support physique sur lequel il est consulté (type de terminal, système d'exploitation..).

Afin de mettre en place notre application nous avons proposé une architecture système basé sur le modèle présenté dans le schéma suivant.



Comme il est possible de le voir dans le modèle suivant nous avons choisi de mettre en place une architecture modulaire s'inspirant des architectures WOA (Web Oriented Architecture) avec notamment l'utilisation d'une API REST.

L'objectif est de mettre en place deux parseurs indépendants permettant de récupérer les données des flux linéaires et non linéaires et de procéder à l'injection des informations analysées dans une base de données mutualisant les programmes TV et VOD au sein d'une unique entité. L'API REST permet, par l'intermédiaire de requêtes d'obtenir les données nécessaires à la visualisation des programmes sur le site Web.

Les flux linéaires seront obtenus sous format XML respectant les critères XMLTV et les flux non-linéaires seront accessible via des fichiers aux formats JSON récupérés via des API, point sur lequel nous reviendrons par la suite. Concernant les images des programmes récupérés, nous avons choisi de ne stocker que les liens correspondant nous permettant d'accéder à ces dernières.

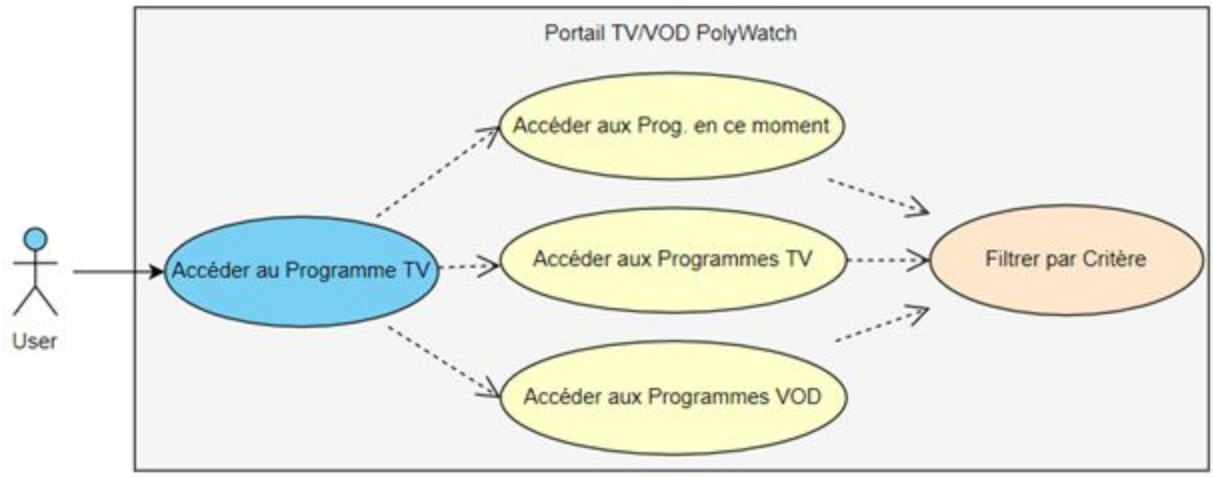
2.2) Caractéristiques des utilisateurs

Notre plateforme Web ne nécessite pas de connexion utilisateur particulière, ce projet ne considère donc qu'un seul profil type utilisateur.

- *User* : Profil classique utilisateur de l'application PolyWatch, on souhaitera rendre sa navigation sur le portail TV/VOD la plus fluide et intuitive possible ce profil regroupant tout type d'utilisateur quelque soit son expérience avec les sites web. LE site ne permet à cet utilisateur que les fonctionnalités propres à la consultation. Cette dernière est accompagnée de la possibilité d'effectuer des filtres sur les résultats obtenus.

Cet utilisateur devra posséder un navigateur web afin d'accéder à l'application. IL est néanmoins important de noter que l'utilisation du site web ne requiert pas de connaissances particulières en informatique.

2.3) Fonctionnalités du système



Notre application web devra permettre aux utilisateurs de la plateforme de réaliser un ensemble de fonctionnalités :

- Traduire le Fichier XML des programmes TV en données injectables dans la base de données et exploitables pour la plateforme Web.

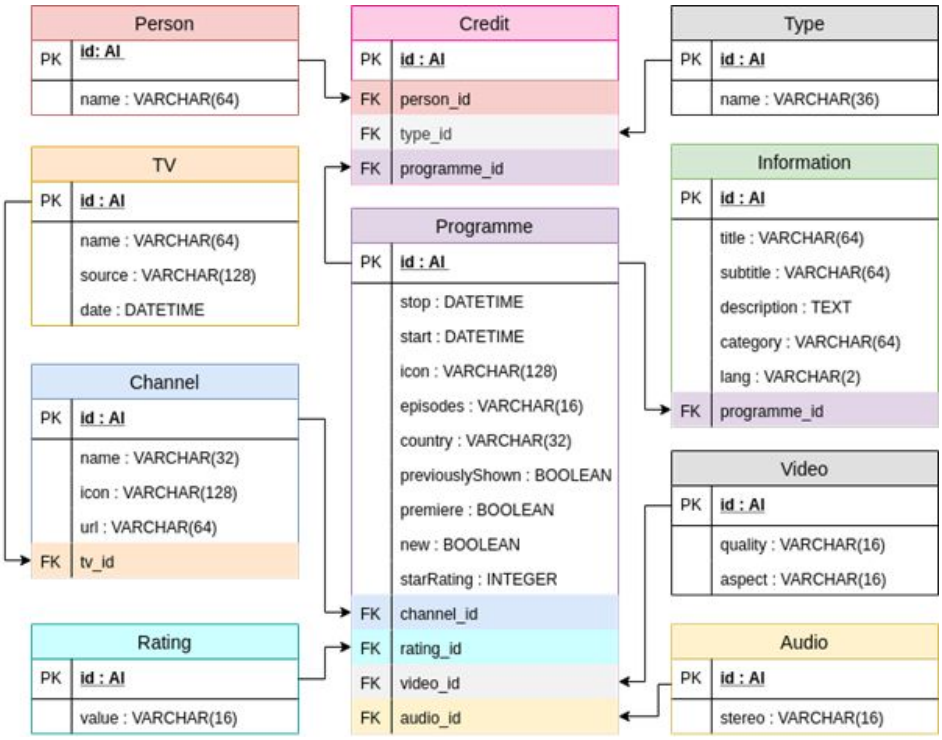
- Traduire le Fichier JSON des programmes Netflix (VOD) récupéré à partir des plusieurs API en données injectables dans la base de données et exploitables pour la plateforme Web.
- Prévoir l'actualisation des données programmes TV et Netflix de la base de données.
- Création des vues composant le portail web : Donner la capacité de visualiser les programmes TV et VOD présents dans la base de données de manière indépendante à travers la création de deux onglets créés dans ce sens.
- Permettre la consultation les programmes TV et VOD les plus « pertinents » : Cette fonctionnalité est ***prioritaire*** par rapport à l'ensemble des fonctionnalités qu'ils nous a été demandé de mettre en place. En effet, cette dernière correspond au nœud de la problématique consistant à mixer les informations des flux linéaires et non linéaires et sera accessible via l'onglet « En ce moment ».
- Créer des filtres de recherche permettant à l'utilisateur d'affiner sa recherche selon des critères qu'il pourra définir.

2.4) Structure générale du système

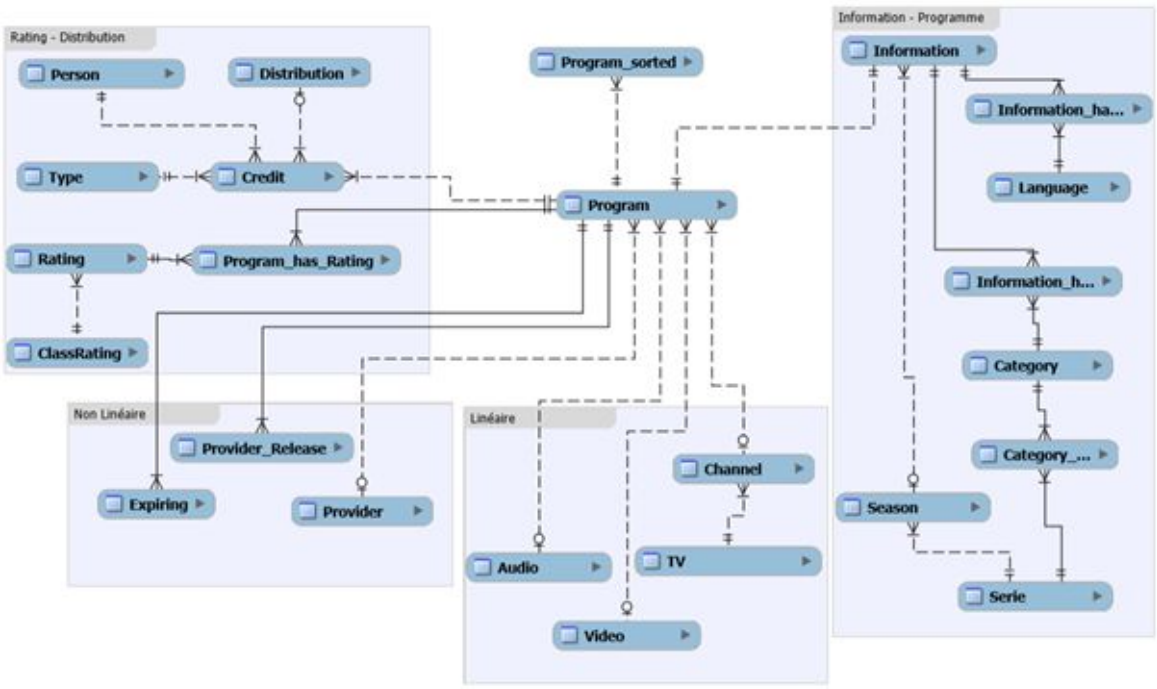
Le projet sera réalisé sur une machine locale et respectera la découpe modulaire précédemment évoquée (Parseur/Base de données & API Rest/ IHM React). Une connexion internet sera nécessaire pour tout utilisateur souhaitant se rendre sur le portail TV/VOD.

Nous nous sommes fortement inspirées du modèle mis en place dans les projets antérieurs de création de guide TV pour mettre en place la structure générale permettant de stocker les données de notre application. Les modèles stockages passés se basant sur la norme XMLTV il était intéressant de se baser dans un premier temps sur cette architecture afin de la faire évoluer vers un modèle correspondant à nos besoins.

Ledit modèle de base étant le suivant :



Nous avons ensuite fait évoluer ce modèle pour lui permettre d'intégrer des programmes VOD avec le souhait de réellement « mixer » les informations TV et VOD. Les nouvelles entités composant le modèle ont donc été ajoutées dans ce sens et regroupées par packages selon le rôle qu'elles occupent.



Nous avons suivi le conseil de mettre en place un système de stockage des données sous un modèle relationnelle ce qui nous a notamment permis comme dit précédemment de s'inspirer de l'architecture d'un projet passé.

Nous disposons donc en entrée de fichiers au format XML correspondant au catalogue TV et de fichiers au format JSON pour les informations concernant le catalogue du distributeur VOD que nous avons choisi à savoir Netflix. Les programmes TV et VOD possèdent la plupart du temps une vignette enregistrée dans les fichiers XML et JSON sous la forme d'un lien URL. Les fichiers seront « parser » au sein du premier module puis injecter dans la base de données inspirée de l'architecture précédente.

Revenons sur le modèle de données adopté pour ce projet afin d'expliquer brièvement les choix réalisé quant à l'organisation des différentes entités composant notre système de stockage.

La classe « Program » représente comme son nom l'indique un programme TV ou VOD et possède un nombre donné d'informations le caractérisant comme un numéro d'épisode ou de saison pour un programme « Série » ou encore les langues dans lesquelles le programme peut être visionné. Le Package « Rating-Distribution » regroupe l'ensemble des personnes (acteurs, réalisateurs ...) ayant participé à la création d'un programme ainsi que les notes attribuées par les critiques.

Les seules distinctions entre TV et VOD du modèle de données sont respectivement présentes dans les packages « Linéaire » et « Non Linéaire ». Il s'agit notamment pour les flux non linéaires de l'identité du fournisseur du programme et de la date de sortie et d'expiration de disponibilité du programme sur la plateforme VOD. Pour les flux TV, nous stockons essentiellement la chaîne sur laquelle est retransmis le programme.

Cette simple structure permet de stocker l'ensemble des informations nécessaires à notre guide média mixant les catalogues TV et VOD.

L'API REST retransmet ensuite les informations composant notre catalogue qui sont stockées dans la base de données par l'intermédiaire de chemins prédéfinis à cet effet. Le dernier module sera composé des vues constituants l'interface utilisateur réalisé à l'aide du Framework ReactJs.

3) Fonctionnalités réalisées

3.1) Lecture du fichier XMLTV

La récupération des données issu du flux linéaire vient de la lecture d'un fichier XML hébergé sur le site XMLTV. Ce site fournit un fichier XML (appelé XMLTV) contenant toutes les informations sur les programmes des 2 prochaines semaines à partir du jour courant et qui inclut l'ensemble des chaînes existantes en France (TNT et payantes).

Après lecture des informations (Chaîne, Programme, ...), un ORM (SQLAlchemy) est utilisé dans la gestion de la base de données SQL afin de faire les insertions et les mises à jour de tel sorte à optimiser les requêtes et à avoir une meilleure lisibilité dans le code.

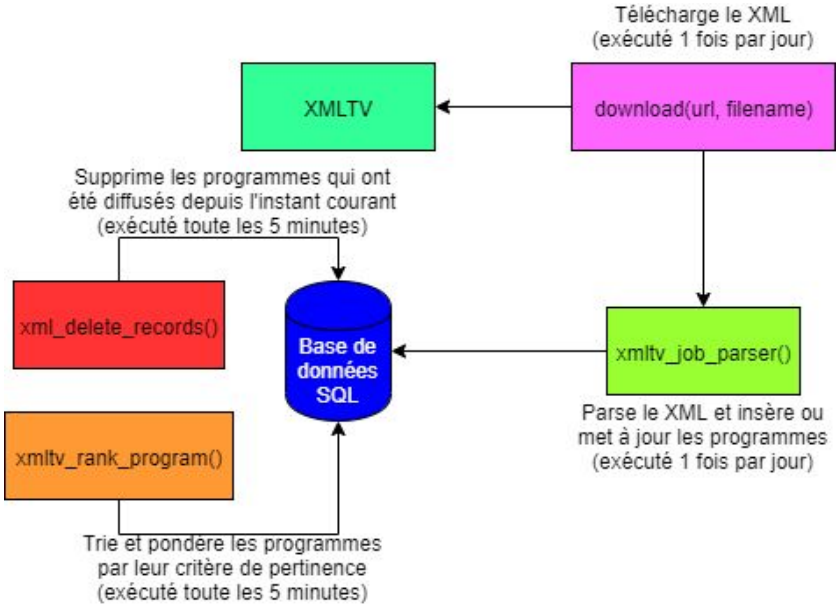
La lecture se fait grâce à la bibliothèque ElementTree de Python, utile pour parcourir facilement un document XML qui nous permet d'évaluer des expressions et récupérer le résultat.

Le parsing fonctionne avec deux fonctions, une qui télécharge le contenu download(url, filename) et une autre qui parse le fichier et insert / update la base de données xmltv_job_parser().

Afin d'éviter d'avoir une base de données qui grossit de façon exponentielle avec les ajouts de programmes, une fonction xmltv_delete_records() a été créée afin de supprimer les programmes qui ont terminé leur diffusion depuis l'instant t (toute les 5 minutes) de la base de données.

Enfin, une autre fonction permet de trier et mettre à jour la table de pertinence des programmes (toute les 5 minutes) par rapport à la pertinence des programme en cours (basé sur le rapport entre le pourcentage commencé par rapport à la durée du programme, avec le poids le plus faible comme le plus pertinent). Les programmes venant de la TNT ont une pondération plus faible car nous considérons qu'il est plus pertinent pour l'utilisateur d'avoir les recommandations en priorité sur des chaînes que chacun possède "gratuitement".

La lecture, l'insertion ou la mise à jour des informations des programmes dans la base de données se fait une fois par jours compte tenu de la rigidité des programmes dans leur programmation et de l'amplitude hebdomadaire du XMLTV. Cette programmation se fait grâce à la crontab du système.



3.2) Récupération des données Netflix

Pour la récupération des flux de type non linéaires, nous avons choisi de nous concentrer sur une seule plateforme VOD : Netflix. Il a fallu trouver une source de données nous permettant de récupérer le catalogue Netflix en français ainsi que les dernières nouveautés et les films qui expirent du catalogue. L'API officielle de Netflix n'étant plus ouverte au public depuis fin 2014, nous avons cherché des API non officielles similaires permettant de répondre à notre besoin.

Nous avons finalement retenu 3 API, chacune répondant à un besoin précis :

- CaptainWatch : API française qui référence le catalogue Netflix France. Cette API permet de récupérer la liste des programmes de Netflix France.
- TMDb (The Movie Database) : Base de données qui permet d'obtenir un grand nombre d'informations sur les films et les séries. Cette API permet de récupérer les détails de chaque programme Netflix.
- uNoGS : API non officielle Netflix payante qui recense des informations sur le catalogue Netflix de chaque pays. Cette API permet de récupérer la liste des nouveautés et des films qui expirent.

L'ensemble de ces API recensent les programmes à l'aide d'un id universel, l'id IMDb (Internet Movie Database). C'est donc cet id qui va permettre de croiser les informations entre les différentes données que nous récupérons des API.

De manière similaire à la lecture du fichier XMLTV, les données Netflix sont parsées et insérées dans la base de données à l'aide d'un script Python et de l'ORM SqlAlchemy. Cependant, comme le flux Netflix est un flux non linéaire, les informations du catalogue ne sont récupérées que toutes les 24h.

3.3) Gestion des images

Dans le fichier XMLTV et dans les API que nous utilisons pour la partie Netflix, les programmes et chaînes utilisent des images qui sont sous forme de lien url. Nous avons souhaité garder ce format dans la base de données afin de ne pas avoir l'inconvénient de devoir gérer les images (taille, poids, ...).

3.4) Générateur de contenus :

3.5) Mise en place de l'architecture back end

Concernant la partie back end, une REST API NodeJS a été mise en place. C'est un serveur qui met à disposition des données au format JSON qui sont contenues dans la base de données afin qu'elles puissent être consommées par un site web par exemple.

L'architecture du serveur REST est la suivante :

- 1 fichier "configDB" de configuration de la base de données pour établir la connexion
- 1 fichier "programs" contenant les requêtes SQL sur la base de données
- 1 fichier "ProgramController" contrôleur sur les requêtes SQL
- 1 fichier "router" contenant les routes mises à disposition
- 1 fichier "index" permettant de lancer le serveur

Le serveur REST retourne sur la route "/api/relevantPrograms" des informations sur les programmes TV pertinents, c'est-à-dire ceux qui passent à la télé lorsque la requête est exécutée, et sur les tendances netflix. On peut de plus appliquer à cette route différents filtres :

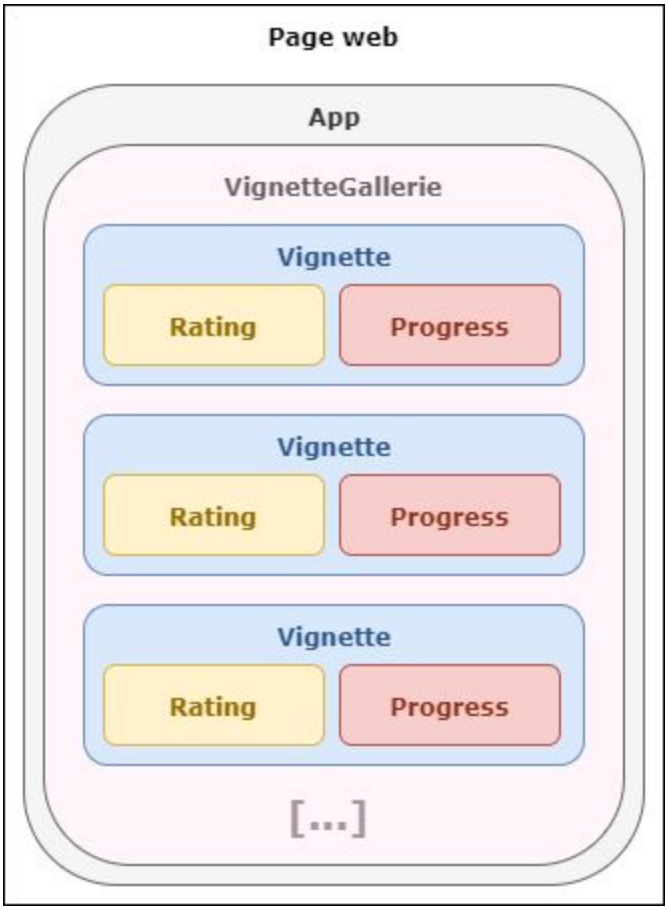
- Filtrer les programmes en affichant les programmes Netflix ou TV
- Filtrer les programmes en passant une catégorie en paramètre
- Filtrer les programmes par type : série/film

Par exemple la route correspondant à la liste des programmes TV qui sont des films est "/api/relevantPrograms?provider=tv&type=film".

3.6) Conception des vues

Notre étude des technologies front-end actuelles nous a orienté vers la bibliothèque JavaScript **ReactJS**, développée par Facebook. De nombreux sites utilisent aujourd'hui cette technologie, on peut citer parmis les plus connus : *Facebook, Netflix, AirBnb, etc*.

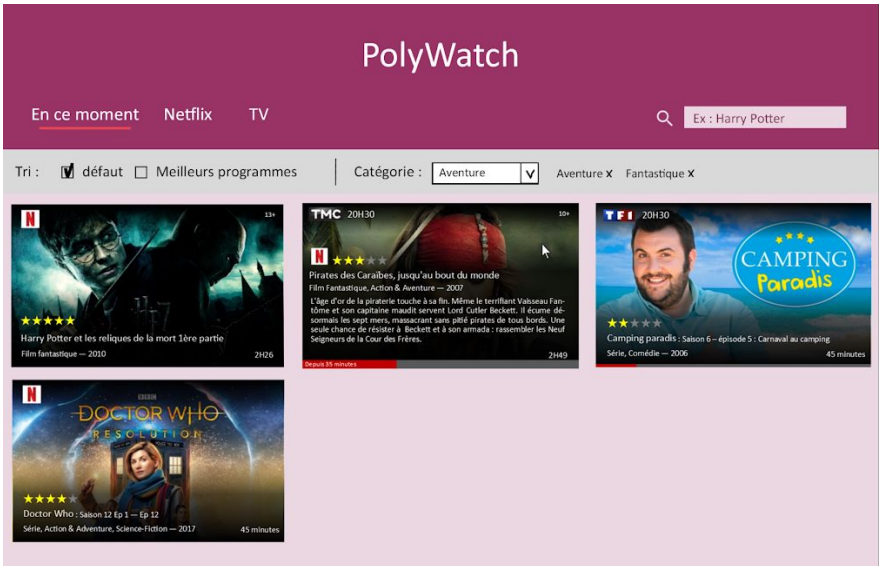
Un projet React s'articule autours de la création de composants, ce qui permet de visualiser clairement l'architecture, comme ici pour notre projet :



Notre structure peut être décrite avec le schéma ci-dessus :

- Notre page web charge le composant principal **App** qui contient le header, champs de recherche et les autres composants
- **VignetteGallerie** contient la liste des vignettes à afficher, et permet d'afficher les détails d'une vignette
- **Vignette** affiche les informations d'un programme
- **Rating** affiche la note du programme (sur 5) sous la forme d'étoiles
- **Progress** affiche une barre de chargement dans le cas où le programme est en cours de diffusion

Notre site ne comprend qu'une seule page :



On a ainsi trois vues différentes pour un programme, les deux premières sous la forme de vignette détaillée ou non si on passe la souris dessus :

Puis un mode détaillé qui s'ouvre quand on clique sur la vignette :

4) Tests de performances

Après mesure des performances du temps de parsing pour les différents flux :

- Base de données à vide :
  - Parser XMLTV :
    - 1 % réalisé ≈ $20\ secondes$, soit 100% ≈ $33\ minutes\ 19\ secondes$
  - Parser Flux Netflix
    - En l'état, avec la taille actuelle du catalogue Netflix France (3391 programmes), ≈ $4\ heures\ 30\ minutes$. Cette valeur peut varier en fonction de la disponibilité des API et de la qualité de la connexion internet, et sera amenée à évoluer avec la taille du catalogue.
- Base de données déjà remplie :
  - Parser XMLTV :
    - 1 % réalisé ≈ $15\ secondes$, soit 100% ≈ $25\ minutes$
  - Parser Flux Netflix
    - Au maximum $15\ minutes$ sont nécessaires pour parser les nouveautés chaque jour.

Cela permet le constat qu'il faut mettre en maintenance le site web (indisponibilité) pendant un temps maximum de 5 heures à l'initialisation et de environ 25 minutes par jours afin d'avoir une grande fiabilité dans les propositions de programmes, que ce soit flux linéaire ou non linéaire.

5) Conclusion

5.1) Conclusion générale

Mettre en place un guide média mutualisant les données de programmes TV et VOD nous a permis de parcourir différents aspects des systèmes d'informations notamment la mise en place d'algorithme pertinence de mise à disposition de l'information.

Le module correspondant à la réalisation des parser XMLTV/JSON VOD et à l'injection des données récupérées dans notre base a été réalisé avec succès. Une automatisation de récupération de ces flux a également été mis en place pour la mise à jour des informations contenues dans la base de données.

Nous sommes également parvenus à mettre en place un système de stockage des données relationnelles permettant la représentation de programme TV et VOD sous une unique représentation de données s'inspirant notamment du modèle XMLTV.

Le portail mis en place par notre équipe récupère les données par l'intermédiaire d'un API REST et permet leur visualisation par l'intermédiaire d'une interface utilisateur utilisant le Framework ReactJs. Le guide média est donc consultable de manière simple, fluide et efficace depuis n'importe quelle plateforme possédant une connexion internet et un navigateur web.

L'objectif était essentiellement de mettre en place un système de données mêlant programmes TV et VOD et de travailler sur un algorithme permettant de faire ressortir les programmes pertinents selon certains critères. On peut donc dire à l'issu de ce projet que cet objectif est rempli.

La création d'un serveur dédié est primordiale pour la viabilité de notre application étant donné les temps relativement importants de chargement des catalogues TV et VOD.

5.2) Améliorations possibles

Compte tenu de la contrainte de temps de réalisation fixé pour ce projet, nous avons pensé à des optimisations et améliorations pouvant être apportés à notre portail TV/VOD.

- Choix d'API pour la récupération des flux Netflix : Comme nous avons pu l'expliquer dans ce rapport nous avons eu des difficultés à trouver une API permettant de récupérer l'ensemble de informations utiles concernant les programmes Netflix pour notre application. Nous avons pu résoudre ce problème en partie en faisant appel à plusieurs API. Il serait néanmoins intéressant d'étudier de nouvelles solutions.

- Conception et ajout de filtres de recherche : Des filtres permettant à l'utilisateur d'affiner ses recherches sur notre portail TV/VOD sont actuellement présents. On peut néanmoins imaginer que davantage de filtres pourraient contribuer à aider l'utilisateur dans ses recherches. Cet ajout prendrait la forme de nouvelles requêtes SQL à intégrer à l'API.

- Mise en place d'un meilleur algorithme de pertinence : L'algorithme actuel prend en compte l'avancée de diffusion du programme TV par rapport à l'heure actuel ainsi que la popularité de la chaîne diffusant ce programme (flux linéaire). Concernant les programmes VOD, la pertinence est pour le moment évaluée en fonction des nouveautés sorties sur la plateforme de distribution VOD.

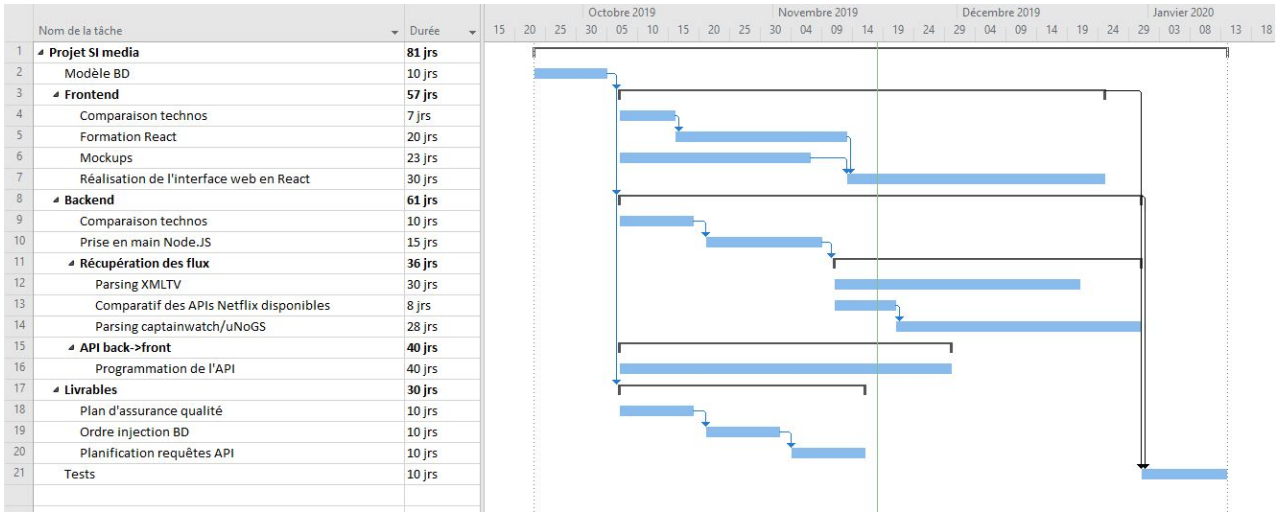
- Analyse ergonomie application, polir l'affichage : Dans l'optique de proposer une application toujours plus « user friendly », il serait réellement intéressant de réfléchir à la meilleure manière de mettre en valeur les informations les plus pertinentes.

6) Annexes

6.1) Planification

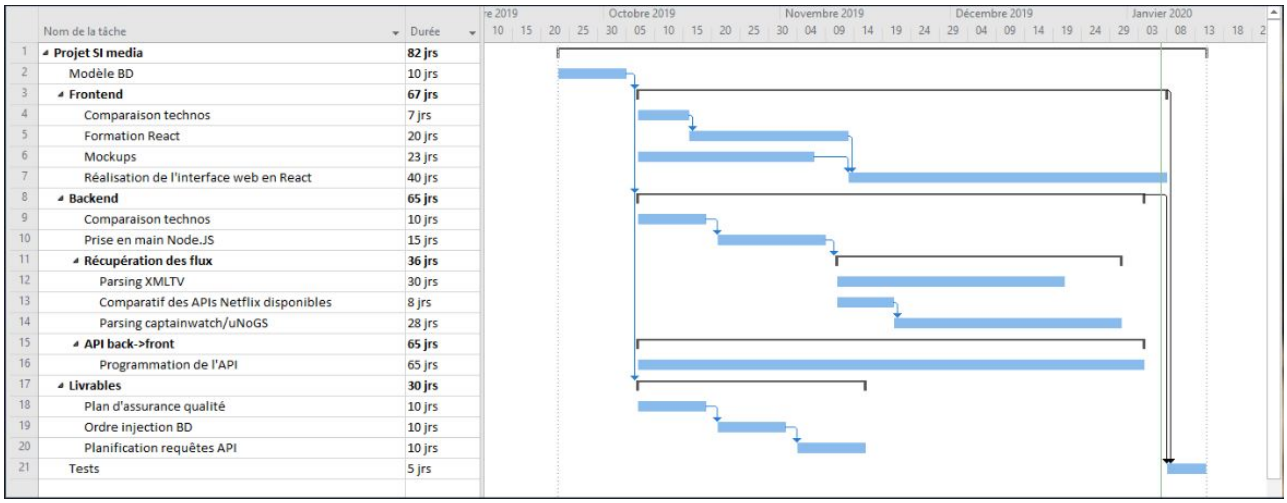
6.1.1) Gantt prévisionnel

Concernant la partie planification de notre projet un premier Gantt prévisionnel a été réalisé.



	Nom de la tâche	Durée
1	**Projet SI media**	**81 jrs**
2	Modèle BD	10 jrs
3	**Frontend**	**57 jrs**
4	Comparaison technos	7 jrs
5	Formation React	20 jrs
6	Mockups	23 jrs
7	Réalisation de l'interface web en React	30 jrs
8	**Backend**	**61 jrs**
9	Comparaison technos	10 jrs
10	Prise en main Node.JS	15 jrs
11	**Récupération des flux**	**36 jrs**
12	Parsing XMLTV	30 jrs
13	Comparatif des APIs Netflix disponibles	8 jrs
14	Parsing captainwatch/uNoGS	28 jrs
15	**API back->front**	**40 jrs**
16	Programmation de l'API	40 jrs
17	**Livrables**	**30 jrs**
18	Plan d'assurance qualité	10 jrs
19	Ordre injection BD	10 jrs
20	Planification requêtes API	10 jrs
21	Tests	10 jrs

Le diagramme de Gantt final comportant quelques modifications sur la fin de projet est le suivant.

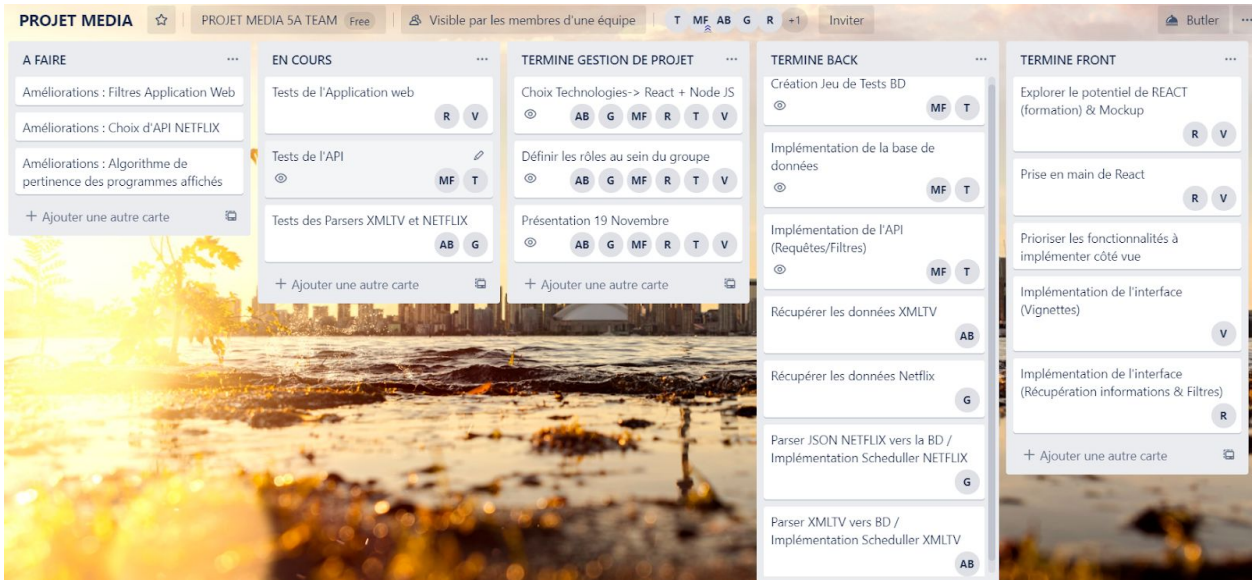


	Nom de la tâche	Durée
1	**Projet SI media**	**82 jrs**
2	Modèle BD	10 jrs
3	**Frontend**	**67 jrs**
4	Comparaison technos	7 jrs
5	Formation React	20 jrs
6	Mockups	23 jrs
7	Réalisation de l'interface web en React	40 jrs
8	**Backend**	**65 jrs**
9	Comparaison technos	10 jrs
10	Prise en main Node.JS	15 jrs
11	**Récupération des flux**	**36 jrs**
12	Parsing XMLTV	30 jrs
13	Comparatif des APIs Netflix disponibles	8 jrs
14	Parsing captainwatch/uNoGS	28 jrs
15	**API back->front**	**65 jrs**
16	Programmation de l'API	65 jrs
17	**Livrables**	**30 jrs**
18	Plan d'assurance qualité	10 jrs
19	Ordre injection BD	10 jrs
20	Planification requêtes API	10 jrs
21	Tests	5 jrs

Certaines tâches n'ont pas pu être traités ou réalisés dans les temps, notamment à cause du retard pris dans le développement de l'API et des vues utilisateur.

6.1.2) Gestion des tâches

Le Trello ci-dessous représente l'évolution du projet et des tâches effectuées.



Le tableau ci-dessous résume quel(s) personne(s) ont été affectée(s) ou ont contribué aux différentes parties mentionnées dans notre projet GitLab :

Tâche	Personne(s) affectée(s)	Personne(s) ayant réalisée(s)
Gestion de Projet	Thomas	Alexandre / Thomas / Guillaume / Martin
Création/Administration de la base de données	Thomas	Thomas
Alimentation de la base de données	Alexandre / Guillaume	Alexandre / Guillaume
Intégration des composants	Groupe entier	-
Gestion du back-end (API)	Martin / Thomas	Martin / Thomas
Lecture du fichier XMLTV	Alexandre	Alexandre
Lecture des flux Netflix	Guillaume	Guillaume

Développement / Intégration des vues	Rémi / Valentin	Rémi / Valentin