

Table des matière

- Animanga
 - Table des versions
 - Introduction
 - Résumé de l'énoncé
 - Organisation
 - Livrable
 - Matériel et logiques à disposition
 - Descriptif complet du projet
 - Sitemap
 - Description succincte du contenu des pages du site
 - Méthodologie
 - 1. S'informer
 - 2. Planifier
 - 3. Décider
 - 4. Réaliser
 - 5. Contrôler
 - 6. Évaluer
 - Planification
 - Product backlog
 - Diagramme de Gantt
 - Implémentation
 - Base de données
 - Structure
 - Classes (Python)
 - API interne
 - Librairies et outils externes
 - Pip et NPM
 - Flask
 - Jinja
 - Flask-Login
 - Flask-Swagger
 - MySQL Connector/Python
 - SASS
 - Swagger
 - jQuery UI
 - ESLint
 - Pylint
 - Git
 - Plans de test et tests
 - Périmètre des tests

- Environnement
- Scénarios
- Évolution des tests
- Bibliographie
- Glossaire
- Conclusion
 - Difficultés majeures rencontrées
 - Améliorations possibles
 - Bilan personnel
 - Remerciements

Table des versions

N° de version	Date	Auteur	Modifications
0.1	2020-05-25	Tanguy Cavagna <tanguy.cvgn@eduge.ch>	Création de la base de la documentation
0.2	2020-05-26	Tanguy Cavagna <tanguy.cvgn@eduge.ch>	Ajout de la partie <i>implémentation</i> et modification des <i>user stories</i> et <i>tests</i> .

Introduction

Ce projet a été réalisé dans le cadre du *Travail Pratique Individuel* (TPI) durant la session de mai - juin 2020. Il a pour but de valider les compétences acquises tout au long de la formation *Informaticien CFC* de l'école du CFPT-Informatique au Petit-Lancy.

Animanga est une application web écrite en Python permettant aux utilisateurs de faire leur propre bibliothèque d'anime. Pour ce faire, l'utilisateur à la possibilité de créer ses propres listes afin de correctement organisé sa bibliothèque, mettre des animes en tant que favoris, et rechercher les animes qu'il voudrait ajouter à sa collection directement depuis cette application.

Résumé de l'énoncé

Les informations suivantes sont extraites du cahier des charges du TPI.

Organisation

Élève	Maître d'apprentissage	Experts
Tanguy Cavagna <tanguy.cvgn@eduge.ch>	Pascal Bonvin <edu-bonvin@eduge.ch>	Nicolas Terrond <nicolas.terrond@sig-ge.ch> Robin Bouille <robin.bouille@gmail.com>

Livrable

Pour les experts et le formateur	Pour le formateur
Planning réel détaillé du projet	Accès au GitHub
Documentation du projet contenant le code source au format PDF	
Journal de bord	
Résumé du TPI (1 page A4)	

Matériel et logiques à disposition

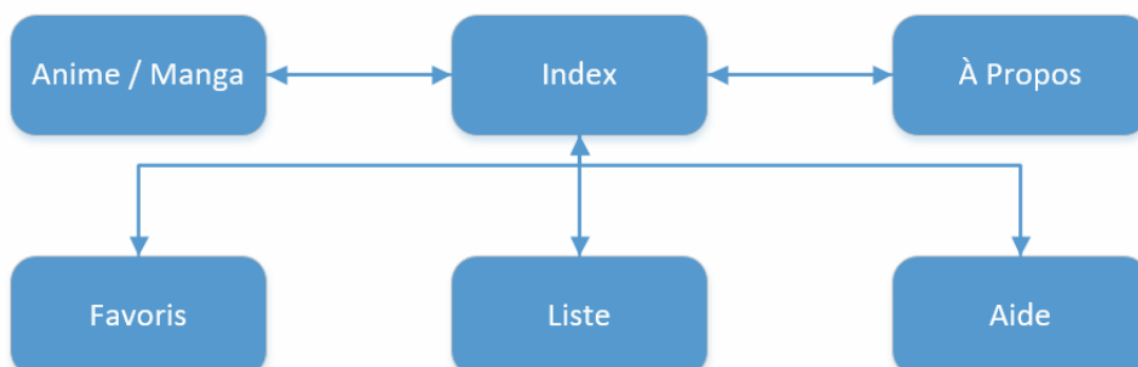
- Un PC standard école, 2 écrans
- Pycharm
- Netbeans
- Suite office
- MySQL, Sqlite3, Flask, connexion internet

Descriptif complet du projet

Lorsqu'il s'agit de réaliser un site web, la tradition de l'école d'informatique encourage l'utilisation de PHP et Mysql. Dans le cas de ce diplôme, il s'agit de réaliser un site local avec Python Flask et de gérer les données dans une base de données Sqlite3. De plus la base de donnée locale est synchronisée unidirectionnelement avec une base de donnée Mysql sur un serveur distant. L'ambition de ce projet est de démontrer qu'il est possible de créer un application WEB sans passer par l'installation d'un serveur apache et d'une base données Mysql. A noter que le candidat est un élève brillant qui maîtrise le langage Python.

Les données initiales qui permettront de remplir la base de données sont accessibles sur github au format json (<https://github.com/manami-project/anime-offline-database>). Elles devront être converties et synchronisées dans la base de données locale qui reste à déterminer.

Sitemap



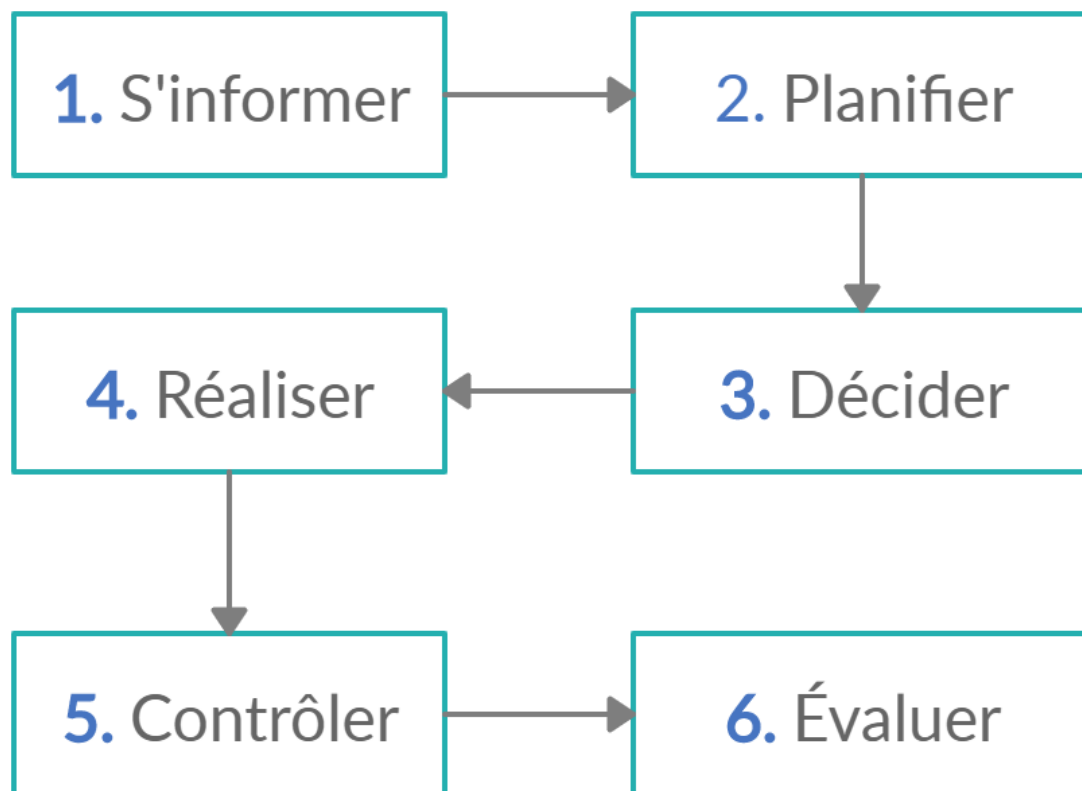
Description succincte du contenu des pages du site

- La page index permet d'avoir le champ de recherche, un bouton "aléatoire", ainsi que les favoris de l'utilisateur et son flux d'activité si connecter.

- La page à propos contient toutes les informations concernant le site ainsi que les librairies utilisées.
- La page connexion permet simplement de se connecter.
- La page inscription, de s'inscrire.
- La page anime / manga permet de voir les informations / actions sur un anime / manga sélectionner (rediriger sur cette page lorsque l'on clique sur un anime / manga sur la page index après une recherche).
- La page aide contient l'aide du site.
- La page profil contient les informations de l'utilisateur ainsi que les listes personnelles, dans lesquelles des animes / manga peuvent être ajoutés, ainsi que leur contenu.
- La page favoris permet de modifier l'ordre des animes / manga mis en favoris ainsi que de les retirer de la liste des favoris.
- La page liste contient un CRUD sur les listes personnelles de l'utilisateur.

Méthodologie

Pour pouvoir planifier correctement ce projet, j'ai décidé d'utiliser la méthode en 6 étapes, décrite ci-dessous :



1. S'informer

La première étape est utile pour pouvoir comprendre le projet dans son ensemble et comprendre toutes les fonctionnalités nécessaires. Il est aussi indispensable de demander d'éclaircir tous les points flous de l'énoncé.

2. Planifier

Le fait de planifier le projet permet de séparer les tâches et de définir des priorités. ses dernières sont les suivantes : ☹ *Bloquant* , ☠ *Critique* , ⚠ *Important* , ? *Secondaire* .

Pour représenter le planning nous avons utilisé un diagramme de Gantt. Ce type de diagramme permet de visualiser très correctement la progression quotidienne ainsi que les différences entre les prévisions et le réel.

3. Décider

Cette partie nous permet de pouvoir se lancer dans la réalisation du projet. S'il nous reste des points en suspens, c'est le moment de prendre une décision et de se jeter à l'eau une bonne fois pour toute.

4. Réaliser

Nous pouvons enfin nous lancer dans l'implémentation de toutes les fonctionnalités à développer ainsi que la rédaction de la documentation.

5. Contrôler

Pour valider cette étape, nous avons tester chacune des fonctionnalités indépendamment des autres pour correctement vérifier leur fonctionnement dans différents cas d'usage.

Une fois l'application terminée, nous avons pu tester son bon fonctionnement sur plusieurs navigateurs différents pour bien être sûre que tout fonctionne comme prévu dans n'importe quel cas d'utilisation.

6. Évaluer

Une fois toutes les étapes précédentes achevées, nous avons pu nous lancer dans ce qui peut sembler le plus complexe. Nous avons fait une rétrospective de tout ce que nous avons fait avec un regard critique afin de chercher des points sur lesquels nous pourrions nous améliorer par la suite. Pour ce faire, nous avons une section dédiée dans le journal de bord répertoriant les problèmes rencontrés ainsi que les solutions trouvées pour ces derniers. Une conclusion est aussi présente à la fin de ce rapport servant de bilan final au projet.

Planification

Product backlog

Nom	S1 : Inscription à Animanga
Description (<i>user story</i>)	En tant qu'utilisateur non connecté, je peux me créer un compte afin de pouvoir accéder au site.
Critère d'acceptation	Les tests 1.1 et 1.10 passent.
Priorité	☹ Bloquant

Nom	S2 : Connexion à Animanga
Description (<i>user story</i>)	En tant qu'utilisateur non connecté, je peux me connecter afin de pouvoir accéder au site.
Critère d'acceptation	Les tests 2.1 et 2.6 passent.
Priorité	🕒 Bloquant

Nom	S3 : Importation des animes
Description (<i>user story</i>)	En tant qu'utilisateur connecté, je peux écraser les animes avec un nouveau set de données.
Critère d'acceptation	Le test 3.1 passe.
Priorité	🚨 Critique

Nom	S4 : Rechercher des animes
Description (<i>user story</i>)	En tant qu'utilisateur connecté, je peux effectuée une recherche afin d'ajouter des animes dans mes listes personnelles ou de les mettre en tant que <i>favoris</i> .
Critère d'acceptation	Les tests 4.1 et 4.2 passent.
Priorité	🚨 Critique

Nom	S5 : Affichage de la carte d'un anime
Description (<i>user story</i>)	En tant qu'utilisateur connecté, je peux cliquer sur le titre d'un anime présent dans les résultats de ma précédente recherche afin d'accéder à ses informations.
Critère d'acceptation	Le test 5.1 passe.
Priorité	📌 Important

Nom	S6 : Mise à jour d'un anime
Description (<i>user story</i>)	En tant qu'utilisateur connecté, je peux mettre à jour le statut, l'appartenance à une liste personnel ainsi que le statut de favoris d'un anime.
Critère d'acceptation	Les tests 6.1 à 6.6 passent.
Priorité	📌 Important

Nom	S7 : Affichage du profile
Description (user story)	En tant qu'utilisateur connecté, je peux avoir accès à ma page de profile afin de pouvoir voir les statistiques et favoris. Il est également possible de voir la page de profile d'autre utilisateur du site.
Critère d'acceptation	Le test 7.1 passe.
Priorité	🚩 Important

Nom	S8 : Affichage des listes
Description (user story)	En tant qu'utilisateur connecté, je peux avoir accès à ma page de listes afin de voir toutes mes listes et leur contenu. Il est également possible de voir les listes d'autre utilisateur du site.
Critère d'acceptation	Le test 8.1 passe.
Priorité	🚩 Important

Nom	S9 : Gestion des listes
Description (user story)	En tant qu'utilisateur connecté, je peux gérer mes propres listes pour en ajouter, en supprimer, ou modifier leur nom.
Critère d'acceptation	Les tests 9.1 et 9.2 passent.
Priorité	🚩 Important

Nom	S10 : Affichage des favoris
Description (user story)	En tant qu'utilisateur connecté, je peux avoir accès à ma page favoris afin de voir tout mes favoris. Il est également possible de voir les favoris d'autre utilisateur du site.
Critère d'acceptation	Les test 10.1 et 10.2 passent.
Priorité	🚩 Important

Nom	S11 : Gestion des favoris
Description (user story)	En tant qu'utilisateur connecté, je peux organiser l'ordre de mes favoris selon mes envies.
Critère d'acceptation	Les tests 11.1 et 11.2 passent.
Priorité	🚩 Important

Nom	S12 : Affichage de la <i>landing page</i>
Description (<i>user story</i>)	En tant qu'utilisateur non connecté, je n'ai ni accès aux animes ni aux listes. La barre de navigation m'affiche un lien pour me connecter et un autre pour m'inscrire.
Critère d'acceptation	Le test 12.1 passe.
Priorité	? Secondaire

Nom	S13 : Utilisation d'un git
Description (<i>user story</i>)	En tant que développeur, je dois pouvoir faire du versionnage de code source et pouvoir accéder à un dépôt distant Github.
Critère d'acceptation	Le dépôt git local est configuré correctement et le lien sur le dépôt distant à été bien fait.
Priorité	⊖ Bloquant

Nom	S14 : Configuration de la base de données
Description (<i>user story</i>)	En tant que développeur, je dois pouvoir utiliser une base de données Sqlite3 et MySQL ayant un modèle identique à celui donné dans l'énoncé. Pour ce faire, j'ai une classe Python me permettant de faire des requêtes sur la base Sqlite3 et une autre classe me permettant de faire des requêtes sur la base MySQL. J'ai aussi un dump de la structure de la base MySQL dans les fichiers statiques de mon application.
Critère d'acceptation	Les tables <code>animes</code> , <code>status</code> , <code>type</code> , <code>url</code> , <code>list</code> , <code>user</code> , <code>list_has_anime</code> , <code>user_has_list</code> et <code>user_has_favorites</code> ont bien été créés et sont utilisables par les contrôleurs dédiés.
Priorité	⊖ Bloquant

Nom	S15 : Synchronisation MySQL Sqlite3
Description (<i>user story</i>)	En tant que développeur, je dois pouvoir synchroniser les bases MySQL et Sqlite3 unidirectionnellement pour créer un backup sur serveur distant.
Critère d'acceptation	Le test 14.1 passe.
Priorité	? Secondaire

Nom	S16 : Configuration Flask
Description (user story)	En tant que développeur, je dois configurer l'application Flask afin d'avoir un site hébergé en local et pouvoir communiquer avec la base de données Sqlite3.
Critère d'acceptation	Une page web s'affiche sur l'url <code>localhost:5000</code> .
Priorité	🚫 Bloquant

Nom	S17: Vérifications syntaxique
Description (user story)	En tant que développeur, je peux lancé la commande <code>npm run lint</code> pour vérifier la syntaxe, basé sur le preset Airbnb, des fichiers JavaScript, et la commande <code>python3 -m pylint --output-format=colored</code> pour vérifier la syntaxe des fichiers Python, basé sur les conventions PEP8.
Critère d'acceptation	Les test 12.1 et 12.2 passent.
Priorité	🚫 Bloquant

Nom	S18: Affichage des activités
Description (user story)	En tant qu'utilisateur connecté, je vois mon fil d'actualité contenant le temps écoulé depuis l'ajout d'un favoris et l'ajout d'un anime dans une liste.
Critère d'acceptation	Le test 15.1 passe.
Priorité	🔍 Secondaire

Diagramme de Gantt

Jour	J1 lu.25	J2 ma.26	J3 me.27	J4 je.28	J5 ve.29	J6 sa.2	J7 di.3	J8 je.4	J9 ve.5	J10 sa.6	J11 di.7
Lecture de l'énoncé	Yellow										
Rédaction backlog	Yellow										
Rédaction scénarios		Yellow	Red								
Rédaction planning		Yellow									
S13 : Utilisation d'un git		Yellow									
S14 : Configuration de la base de données			Yellow								
S16 : Configuration de Flask			Yellow								
S3 : Importation des données		Yellow	Yellow								
S1 : Inscription à Animanga			Yellow								
S2 : Connexion à Animanga			Yellow	Green							
S12 : Affichage de la landing page			Red	Yellow							
S4 : Rechercher des animes				Yellow							

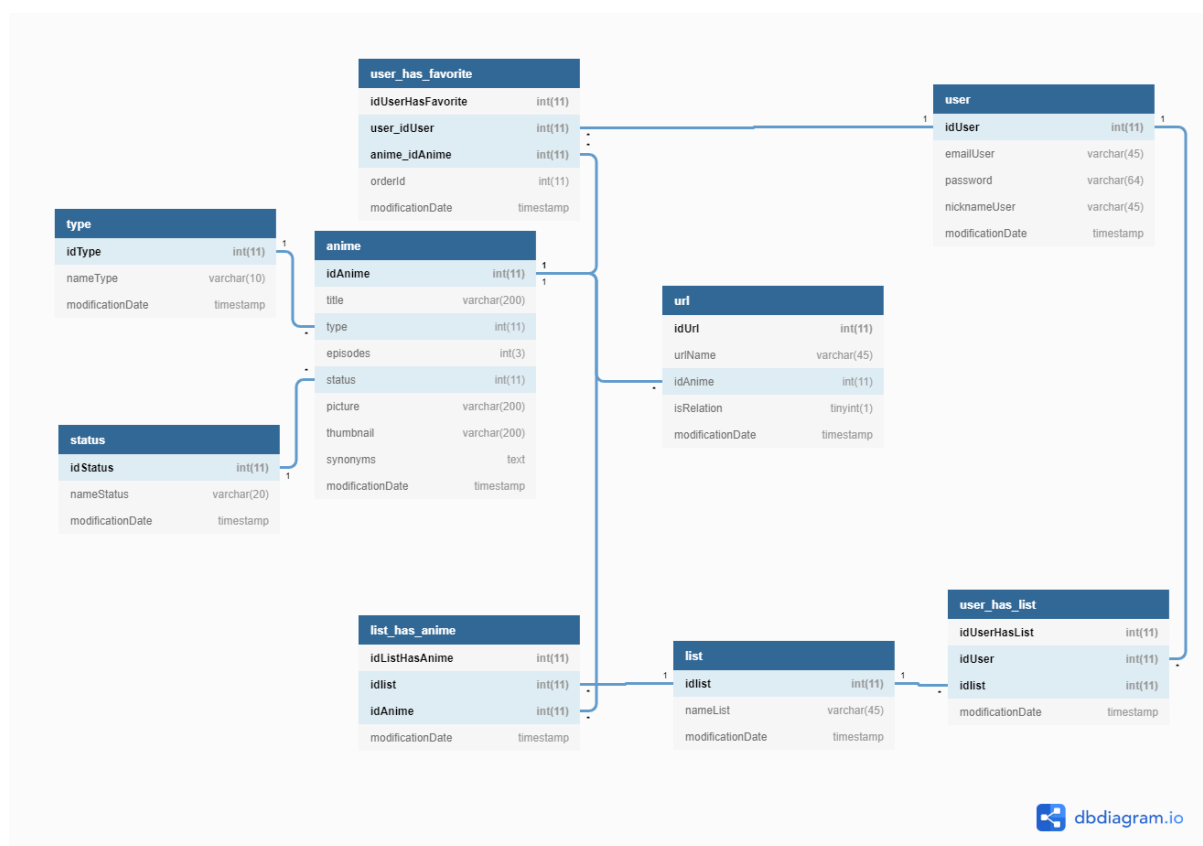
Jour	J1 lu.25	J2 ma.26	J3 me.27	J4 je.28	J5 ve.29	J6 sa.30	J7 di.31	J8 je.4	J9 ve.5	J10 sa.6	J11 di.7
S5 : Affichage de la carte de l'anime											
S6 : Mise à jour de l'anime											
S7 : Affichage du profil											
S10 : Affichage des favoris											
S8 : Affichage des listes											
S9 : Gestion des listes											
S11 : Organisation des favoris											
S18 : Affichage des activités											
S15 : Synchronisation MySQL Sqlite3											
S17 : Vérifications syntaxique											
Tests en profondeur et corrections des bugs											
Tenir à jour la documentation											
Résumé du TPI											
Finalisation / Impression											
Tenue du journal de bord											

Implémentation

Base de données

Le projet de TPI contient 2 types de base de données différents. La base principale est en Sqlite3 et la base de backup distante est en MySQL. Ces deux bases doivent pouvoir stocker les utilisateurs, les animes, les listes personnelles des utilisateurs, ainsi que les favoris des utilisateurs.

Voici le modèle réalisé :



Dictionnaire de données

anime

Colonne	Type	Null
idAnime	int(11)	Non
title	varchar(200)	Non
<i>#type</i>	int(11)	Non
episodes	int(3)	Non
<i>#status</i>	int(11)	Non
picture	varchar(200)	Non
thumbnail	varchar(200)	Non
synonyms	text	Oui
modificationDate	timestamp	Non

status

Colonne	Type	Null
idStatus	int(11)	Non
nameStatus	varchar(20)	Non
modificationDate	timestamp	Non

type

Colonne	Type	Null
idType	int(11)	Non
nameType	varchar(10)	Non
modificationDate	timestamp	Non

url

Colonne	Type	Null
idUrl	int(11)	Non
urlName	varchar(45)	Non
<i>#idAnime</i>	int(11)	Non
isRelation	tinyint(1)	Non
modificationDate	timestamp	Non

list

Colonne	Type	Null
idList	int(11)	Non
nameList	varchar(45)	Non
modificationDate	timestamp	Non

list_has_anime

Colonne	Type	Null
idListHasAnime	int(11)	Non
#idList	int(11)	Non
#idAnime	int(11)	Non
modificationDate	timestamp	Non

user

Colonne	Type	Null
idUser	int(11)	Non
emailUser	varchar(45)	Non
password	varchar(45)	Non
nicknameUser	varchar(45)	Non
modificationDate	timestamp	Non

user_has_list

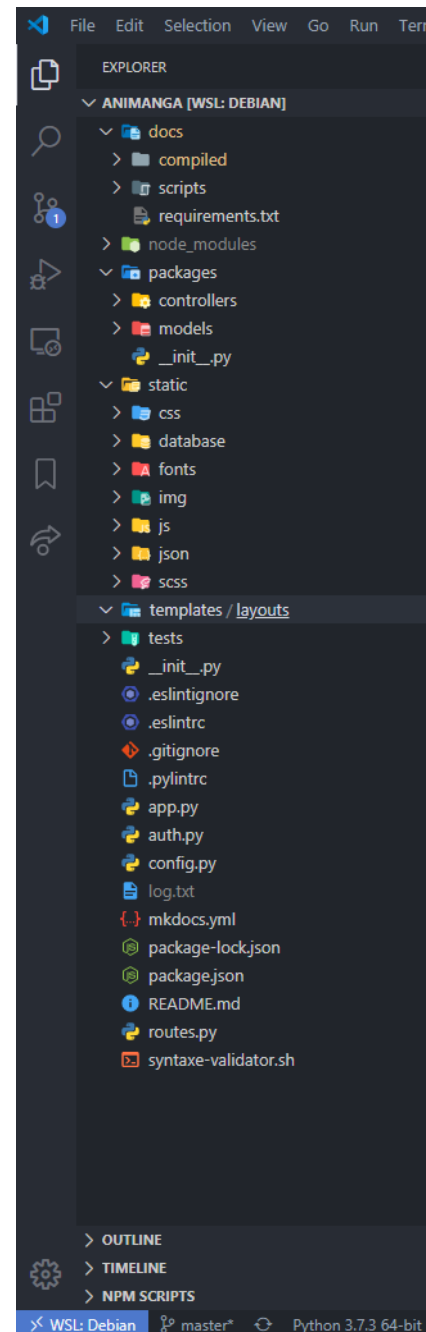
Colonne	Type	Null
idUserHasList	int(11)	Non
#idUser	int(11)	Non
#idList	int(11)	Non
modificationDate	timestamp	Non

user_has_favorite

Colonne	Type	Null
idUserHasFavorite	int(11)	Non
#idUser	int(11)	Non
#idAnime	int(11)	Non
orderId	int(11)	Non
modificationDate	timestamp	Non

Structure

- **/** : Contient tout les fichiers de configuration ainsi que les routes
- **/docs** : Contient la documentation de tout mon projet
 - **/docs/compiled** : Contient la documentation compilée
 - **/docs/scripts** : Contient tout les scripts permettant de créer un fichier unique contenant toute la documentation
- **/node_modules** : Contient les dépendances JavaScript (géré par NPM)
- **/packages** : Contient les fichiers `python`
 - **/packages/controllers** : Contient les contrôleurs
 - **/packages/models** : Contient les modèles de données
- **/static** : Contient les fichiers statiques
 - **/static/css** : Contient les fichiers `css`
 - **/static/database** : Contient la base de données sqlite3
 - **/static/fonts** : Contient les polices
 - **/static/img** : Contient les images / `svg`
 - **/static/js** : Contient les fichiers `javascript`
 - **/static/json** : Contient le fichier `json` contenant tout les animes
 - **/static/scss** : Contient les fichiers `scss`
 - **/static/swagger-components** : Contient les composants pour swagger
- **/templates** : Contient les pages à afficher
 - **/templates/layouts** : Contient le layout générique ainsi que les fichier pouvant être inclus
- **/tests** : Contient le fichier `html` pour Katalon Recorder



Classes (Python)

Pour correctement interagir avec le code Python, et comme demandé dans le point **A20**, j'ai écrit les classes suivantes :

\packages\controllers\ActivitiesController

Classe permettant le contrôle des activités de l'utilisateur

\packages\controllers\AnimeController

Classe permettant le contrôle des animes

\packages\controllers\authentication

Contient les classe permettant la validation des données des formulaires d'authentification

\packages\controllers\ListController

Classe permettant le contrôle des listes

\packages\controllers\logger

Contient la fonction utilisée par tout les `except` afin de log les potentielles erreurs

\packages\controllers\MySQLController

Classe permettant d'interagir avec la base de données MySQL

\packages\controllers\SqliteController

Classe permettant d'interagir avec la base de données Sqlite3

\packages\controllers>StatusController

Classe permettant le contrôle des statuts

\packages\controllers\TypeController

Classe permettant le contrôle des types

\packages\controllers\UserController

Classe permettant le contrôle des utilisateurs

\packages\models\Anime

Modèle représentant un anime

\packages\models>List

Modèle représentant une liste

\packages\models>Status

Modèle représentant un statut

\packages\models\Type

Modèle représentant un type

\packages\models\User

Modèle représentant un utilisateur

API interne

Animanga possède un système d'API interne permettant de faire des actions sur les données depuis le front-end. Voici les différentes url d'entrées :

/get/favorites/

Permet de récupérer tous les favoris de l'utilisateur connecté

/get/favorites/<string:nickname>

Permet de récupérer tous les favoris d'un utilisateur

/set/favorite

Met à jour le statut de favoris d'un anime pour l'utilisateur connecté

/set/list

Met à jour l'association d'un anime à une liste

/delete/defaults

Permet de supprimer l'anime des listes par défauts de l'utilisateur connecté (Complétés, En cours, Planifiés, Abandonnés)

/get/animes

Permet de récupérer les animes d'une liste

/add/list

Permet d'ajouter une nouvelle liste à l'utilisateur connecté

/delete/list

Permet de supprimer une liste de l'utilisateur connecté

/set/list/name

Met à jour le nom d'une liste

/set/favorites-order

Met à jour l'ordre des favoris

/get/activities

Permet de récupérer toutes les activités des dernières 24h de l'utilisateur connecté

Librairies et outils externes

Pip et NPM

Pip et **NPM** sont deux gestionnaires de dépendances que j'ai utilisé pour mon TPI. Pip est le gestionnaire des dépendances Python tandis que NPM est son équivalent pour JavaScript. Ces deux gestionnaires m'ont permis d'inclure toutes les librairies externes que j'avais besoin pour mon TPI. Ceci me permet de ne pas avoir à télécharger manuellement les librairies et à les mettre dans mon projet. Leur utilisation m'a permis de grandement faciliter le développement du TPI et d'avoir des dépendances toujours à jour.



Flask

Flask est un micro framework web écrit en Python. Aucune couche autre que l'hébergement web n'est présente dans ce micro framework. Flask a été créé par **Armin Ronacher**, membre de **Pocoo**, un groupe de développeurs Python



formé en 2004, le 1^{er} avril 2010. J'ai choisi d'utiliser ce framework pour mon TPI car il m'a permis d'aisément :

- Héberger mon site en local ainsi que de pouvoir créer des routes web. Ces dernières sont url écrites dans la barre d'adresse du navigateur. Elles sont utilisées pour éviter de devoir écrire en dure les nom des fichiers à afficher ainsi que de pouvoir exécuter du code avant d'afficher la page à l'utilisateur afin de récupérer des informations nécessaire au bon affichage des informations dynamiques. Un bon exemple d'utilisation est la page d'accueil : si l'utilisateur n'est pas connecté, un fond contenant une image est affiché et la barre de navigation du site ne permet que d'avoir accès à l'accueil, la page à propos, la page de connexion et enfin d'inscription. L'informations comme quoi l'utilisateur n'est pas connecté est récupérée avant que la page soit affiché.
- Configurer le debug de mon site de manière générale. Il est possible de données des paramètres de configuration à l'application Flask afin de facilité le développement. J'ai utilisé ces paramètres pour facilité le rafraichissement des pages dès lors qu'une modification est détectée dans un fichier.

Jinja

Jinja est un moteur de modèle de page web pour Python. Il a été créer par **Armin Ronacher**. Sa syntaxe est relativement identique au moteur de modèle Django mais adaptée pour la syntaxe de Python. Ce moteur de modèle est celui par défaut de **Flask**.



Flask-Login

Flask-Login donne accès à un gestionnaire de sessions pour **Flask**. Il prend en compte les tâches standards comme la connexion, la déconnexion, et l'enregistrement de l'utilisateur en session sur une long période de temps. Dans mon TPI je l'utilise afin de connecter / déconnecter mes utilisateur et pour pouvoir stocker leurs informations en session durant leur utilisation du site.

Flask-Swagger

Flask-Swagger donne accès à une méthode (swagger) qui inspecte les routes de **Flask** contenant une docstring en YAML afin de générer les spécifications nécessaires à **Swagger**.

MySQL Connector/Python

MySQL Connector/Python est une librairie permettant à Python de communiquer avec les serveurs MySQL. Cette librairie est indispensable si l'on veut communiquer avec une base de données MySQL, et elle apporte des avantages tel que la conversion de données entre Python et MySQL. Par exemple, le `datetime` Python et `DATETIME` MySQL.

SASS

SASS est un préprocesseur CSS. Cet outil permet d'étendre la syntaxe du langage CSS afin de pouvoir ajouter de nouvelles fonctionnalités. SASS permet aussi d'avoir un système de variable plus puissant que celui de CSS ainsi qu'un système d'import de fichier plus épuré à mon goût. En effet, il est possible de créer un fichier pour stocké toutes les couleurs sous forme de variables et ensuite importé ce fichier dans la feuille de style principale pour pouvoir utilisé les couleurs n'importe où.



Swagger

Swagger permet de visualiser les url d'une API automatiquement, basé sur les spécifications de chaque url. La générations du visuel est automatique et est optimisé pour une interaction avec le client. J'ai utilisé cet outil afin de visualiser correctement les routes utilisées par mon application afin de récupérer des données.

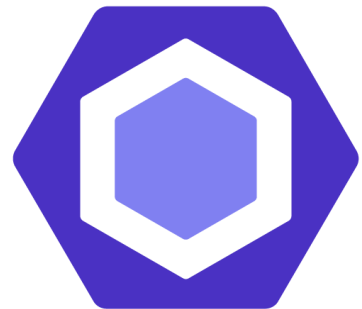


jQuery UI

jQuery UI est un ensemble d'interactions utilisateur, d'effets, de widget, et de thème construit sur la base de jQuery. J'ai utilisé cet outil afin de pouvoir gérer avec facilité la réorganisation des favoris d'un utilisateur. En effet, il est possible de glisser déposé les couvertures des animes présent dans les favoris de l'utilisateur afin de réorganisé l'ordre de ces derniers.

ESLint

ESLint est un outil vérification syntaxique automatique de code. La vérification est basé sur un ensemble de règles définissant la syntaxe à utiliser. Cet outil m'a été utile pour vérifier que mon code était conforme aux normes **Airbnb**, pour éviter d'avoir des morceaux de code potentiellement problématiques ou mal optimiser ou même plus utilisé.



```
.thor/Animanga x + v
(๑_๑) ~/Documents/programming/python/Animanga master npm run lint static/js
npm WARN npm npm does not support Node.js v10.19.0
npm WARN npm You should probably upgrade to a newer version of node as we
npm WARN npm can't make any promises that npm will work with this version.
npm WARN npm Supported releases of Node.js are the latest release of 4, 6, 7, 8, 9.
npm WARN npm You can find the latest version at https://nodejs.org/

> Animanga@1.0.0 lint /home/cavagnat/Documents/programming/python/Animanga
> eslint '**/*.js' --ignore-pattern node_modules/ "static/js"

/home/cavagnat/Documents/programming/python/Animanga/static/js/user-lists-handler.js
304:22 error Missing semicolon semi

x 1 problem (1 error, 0 warnings)
1 error and 0 warnings potentially fixable with the --fix option.

npm ERR! code ELIFECYCLE
npm ERR! errno 1
npm ERR! Animanga@1.0.0 lint: `eslint '**/*.js' --ignore-pattern node_modules/ "static/js"`
npm ERR! Exit status 1
npm ERR!
npm ERR! Failed at the Animanga@1.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /home/cavagnat/.npm/_logs/2020-06-05T07_37_00_008Z-debug.log
x (๑_๑) ~/Documents/programming/python/Animanga master
```

Cas d'utilisation de ESLint. La command `npm run lint static/js` m'indique qu'un point-virgule est manquant à la ligne 93 de mon fichier `user-list-handler.js`

Pylint

Pylint est aussi un outil de vérification syntaxique comme **ESLint**. Cependant, il n'utilise pas de standard créer par la communauté mais les standards officiels de Python, le **PEP8**.



```
.thor/Animanga x + v
(๑_๑) ~/Documents/programming/python/Animanga master python3 -m pylint --output-format=colored packages/controllers
***** Module packages.controllers.SqliteController
packages/controllers/SqliteController.py:9:0: C0411: standard import "import sqlite3" should be placed before "from flask import g" (wrong-import-order)
packages/controllers/SqliteController.py:10:0: C0411: standard import "from typing import Any" should be placed before "from flask import g" (wrong-import-order)
packages/controllers/SqliteController.py:11:0: C0411: standard import "from sqlite3 import Error as SqliteError" should be placed before "from flask import g" (wrong-import-order)

-----
Your code has been rated at 9.96/10 (previous run: 9.96/10, +0.00)

x (๑_๑) ~/Documents/programming/python/Animanga master
```

Cas d'utilisation de Pylint. La command `pylint --output-format=colored packages/controllers/SqliteController.py` m'indique entre autre que des imports ne sont pas bien placés

Git

Git est un outil de gestion de version. Cet outil à été utilisé durant toute la durée de mon TPI afin de garder un historiques des modifications apportées à mon projet ainsi qu'un système de sauvegarde externe sur **Github** en cas de problème technique sur mon ordinateur local.



Plans de test et tests

Périmètre des tests

Pour *Animanga* j'ai mis en place un protocole de test afin que n'importe quel utilisateur puisse naviguer convenablement dans l'application, peu importe son navigateur WEB.

Environnement

Lors de ces tests, j'ai utilisé les navigateurs suivants :

- Mozilla Firefox 76.0.1 (64 bits) sur Windows 10 Entreprise 1903
- Google Chrome 81.0.4044.138 (64 bits) sur Windows 10 Entreprise 1903
- Microsoft Edge Version 81.0.416.72 (64 bits) sur Windows 10 Entreprise 1903

Scénarios

Les scénarios des tests sont détaillés afin que n'importe quelles personne puisse les exécuter. Pour rédiger mes scénarios j'ai utilisé la syntaxe **Gherkin**.

Nom	1.1 Création d'un nouveau compte (informations valides)
User Story	S1 : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: MotDePasse , confirmation: MotDePasse .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon nouveau compte connecté.</p>
Résultats obtenus	Je suis redirigé vers la page d'accueil avec mon nouveau compte connecté.
Statut	✓ OK

Nom	1.2 Création d'un nouveau compte (informations invalides)
User Story	S1 : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: -, pseudo: <i>Katalon</i> , MDP: <i>MotDePasse</i> , confirmation: <i>MotDePasse</i> .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email n'est pas présent.
Statut	✓ OK

Nom	1.3 Création d'un nouveau compte (informations invalides)
User Story	S1 : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: <i>a@b.c</i> , pseudo: <i>Katalon</i> , MDP: <i>MotDePasse</i> , confirmation: <i>MotDePasse</i> .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email fourni est trop court.
Statut	✓ OK

Nom	1.4 Création d'un nouveau compte (informations invalides)
User Story	SI : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: invalide/@recorder.ch , pseudo: Katalon , MDP: MotDePasse , confirmation: MotDePasse .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email fourni n'est pas correct.
Statut	✓ OK

Nom	1.5 Création d'un nouveau compte (informations invalides)
User Story	SI : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: - , MDP: MotDePasse , confirmation: MotDePasse .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que le pseudo n'est pas présent.
Statut	✓ OK

Nom	1.6 Création d'un nouveau compte (informations invalides)
User Story	SI : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: - , confirmation: - .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que le mot de passe n'est pas présent.
Statut	✓ OK

Nom	1.7 Création d'un nouveau compte (informations invalides)
User Story	SI : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: Court , confirmation: Court .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que le mot de passe est trop court.
Statut	✓ OK

Nom	1.8 Création d'un nouveau compte (informations invalides)
User Story	SI : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: - , confirmation: - .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que le mot de passe de confirmation n'est pas présent.
Statut	✓ OK

Nom	1.9 Création d'un nouveau compte (informations invalides)
User Story	S1 : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: MotDePasse , confirmation: MotDePasseDifferent .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que les mots de passes sont différent.
Statut	✓ OK

Nom	1.10 Création d'un nouveau compte (informations invalides)
User Story	S1 : Inscription à Animanga
Situation	<p>Étant donné que je suis un nouvel utilisateur de Animanga, je ne possède pas encore de compte.</p> <p>Quand je clique sur le bouton <i>Inscription</i> , je suis redirigé vers la page d'inscription et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , pseudo: Katalon , MDP: MotDePasse , confirmation: MotDePasseDifferent .</p> <p>Alors, le formulaire est rechargé avec un message d'erreur indiquant ce qu'il ne c'est pas bien passé.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email est déjà utilisé.
Statut	✓ OK

Nom	2.1 Connexion avec un compte existant (informations valides)
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur de Animanga, j'ai déjà un compte à disposition.</p> <p>Quand je clique sur le bouton <i>Connexion</i> , je suis redirigé vers la page de connexion et je rempli le formulaire avec les informations suivantes : email: katalon@recorder.ch , MDP: MotDePasse .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon compte connecté.</p>
Résultats obtenus	Je suis redirigé vers la page d'accueil avec mon nouveau compte connecté.
Statut	✓ OK

Nom	2.2 Connexion avec un compte existant (informations invalides)
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur de Animanga, j'ai déjà un compte à disposition.</p> <p>Quand je clique sur le bouton <i>Connexion</i> , je suis redirigé vers la page de connexion et je rempli le formulaire avec les informations suivantes : email: - , MDP: <i>MotDePasse</i> .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon compte connecté.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email n'est pas présent.
Statut	✓ OK

Nom	2.3 Connexion avec un compte existant (informations invalides)
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur de Animanga, j'ai déjà un compte à disposition.</p> <p>Quand je clique sur le bouton <i>Connexion</i> , je suis redirigé vers la page de connexion et je rempli le formulaire avec les informations suivantes : email: <i>invalide/@recorder.ch</i> , MDP: <i>MotDePasse</i> .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon compte connecté.</p>
Résultats obtenus	Un message s'affiche m'indiquant que l'email est invalide.
Statut	✓ OK



Nom	2.4 Connexion avec un compte existant (informations invalides)
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur de Animanga, j'ai déjà un compte à disposition.</p> <p>Quand je clique sur le bouton <i>Connexion</i> , je suis redirigé vers la page de connexion et je rempli le formulaire avec les informations suivantes : email: <i>katalon@recorder.ch</i> , MDP: - .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon compte connecté.</p>
Résultats obtenus	Un message s'affiche m'indiquant que le mot de passe n'est pas présent.
Statut	✓ OK

Nom	2.5 Connexion avec un compte existant (informations invalides)
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur de Animanga, j'ai déjà un compte à disposition.</p> <p>Quand je clique sur le bouton <i>Connexion</i> , je suis redirigé vers la page de connexion et je rempli le formulaire avec les informations suivantes : email: <i>katalon@recorder.ch</i> , MDP: <i>MotDePasseInexistant</i> .</p> <p>Alors, je suis redirigé sur la page d'accueil avec mon compte connecté.</p>
Résultats obtenus	Un message s'affiche m'indiquant que la combinaison email - mot de passe est invalide.
Statut	✓ OK

Nom	2.6 Déconnexion
User Story	S2 : Connexion à Animanga
Situation	<p>Étant donné que je suis un utilisateur connecté au site.</p> <p>Quand je clique sur le bouton <i>Déconnexion</i> placé dans le dropdown du menu <i>Utilisateur</i> .</p> <p>Alors, je deviens un utilisateur non connecté et je suis redirigé sur la page de connexion.</p>
Résultats obtenus	Je clique sur <i>Utilisateur</i> et <i>Déconnexion</i> . Je ne suis plus connecté et je suis revenue sur la page de connexion.
Statut	✓ OK

Nom	3.1 Importation des animes
User Story	S3 : Importation des animes
Situation	<p>Étant donné que je suis un utilisateur connecté au site.</p> <p>Quand je clique sur le bouton <i>Écraser tous les animes</i> placé dans le dropdown du menu <i>Utilisateur</i> .</p> <p>Alors, j'écrase toutes les données du site relatives aux animes. Cela comprend les animes en eux même, les favoris ainsi que les animes contenu dans les listes.</p>
Résultats obtenus	Je clique sur <i>Utilisateur</i> et <i>Écraser tous les animes</i> . Je suis redirigé vers la page d'accueil et des alertes s'affiche en haut au centre de l'écran indiquant l'état de la mise à jours des animes.
Statut	✓ OK

Nom	4.1 Recherche des animes
User Story	S4 : Recherche des animes
Situation	<p>Étant donné que je suis un utilisateur connecté au site.</p> <p>Quand je clique sur le bouton  placé dans la barre de navigation et que j'écris "k On" dans le champs de recherche de la modale.</p> <p>Alors, je suis redirigé vers la page d'accueil et les résultats de la recherche affiche l'anime "K-ON!".</p>
Résultats obtenus	L'anime "K-ON!" est présent dans la zone de résultat de recherche.
Statut	✓ OK

Nom	4.2 Recherche des animes avec raccourcis
User Story	S4 : Recherche des animes
Situation	<p>Étant donné que je suis un utilisateur connecté au site.</p> <p>Quand je fais le raccourcis clavier  +  et que j'écris "k On" dans le champs de recherche de la modale.</p> <p>Alors, je suis redirigé vers la page d'accueil et les résultats de la recherche affiche l'anime "K-ON!".</p>
Résultats obtenus	L'anime "K-ON!" est présent dans la zone de résultat de recherche.
Statut	✓ OK

Nom	5.1 Affichage de la carte de l'anime
User Story	S5 : Affichage de la carte de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche.</p> <p>Quand je clique sur le titre d'un anime présent dans la zone de résultat de la recherche.</p> <p>Alors, une modale s'affiche contenant l'image de couverture, le titre, le statut de visionnement de l'anime, et les listes personnelles de l'utilisateur.</p>
Résultats obtenus	La modale s'affiche avec le contenu adéquat.
Statut	✓ OK

Nom	6.1 Mise à jour du statut de l'anime sélectionné
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand sélectionne un statut autre que "---".</p> <p>Alors, le combo-box se met à jour avec la nouvelle valeur sélectionnée.</p>
Résultats obtenus	La valeur du combo-box c'est bien mise à jour.
Statut	✓ OK

Nom	6.2 Ajout de l'anime dans une liste personnelle
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand je clique sur une check-box blanche d'une des listes personnelles.</p> <p>Alors, l'état de la check-box se met à jour et elle se colore en bleu. L'anime est maintenant présent dans la liste personnelle.</p>
Résultats obtenus	L'état de la check-box c'est bien mis à jour et est bien coloré en bleu.
Statut	✓ OK

Nom	6.3 Ajout de l'anime dans les favoris
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand je clique sur le cœur blanc pour ajouté au favoris.</p> <p>Alors, le cœur se colore et l'anime se rajoute dans la zone des favoris de la page d'accueil.</p>
Résultats obtenus	Le cœur c'est coloré et l'anime c'est correctement ajouté dans la zone des favoris de la page d'accueil.
Statut	✓ OK


Nom	6.4 Suppression du statut de l'anime
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand sélectionne le statut "---".</p> <p>Alors, le combo-box se met à jour avec la nouvelle valeur sélectionnée et l'anime n'est plus présent dans aucun autre statut.</p>
Résultats obtenus	La valeur du combo-box c'est bien mise à jour et l'anime n'est effectivement plus présent dans les autres statuts.
Statut	✓ OK


Nom	6.5 Suppression de l'anime d'une liste personnelle
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand je clique sur une check-box bleue d'une des listes personnelles.</p> <p>Alors, l'état de la check-box se met à jour et se colore en blanc. L'anime n'est plus présent dans la cette liste personnelle.</p>
Résultats obtenus	L'état de la check-box c'est bien mis à jour et est coloré en blanc.
Statut	✓ OK

Nom	6.6 Suppression de l'anime des favoris
User Story	S6 : Mise à jour de l'anime
Situation	<p>Étant donné que je suis un utilisateur connecté au site sur la page d'accueil ayant fait une recherche et ayant ouvert la modale d'informations d'un anime.</p> <p>Quand je clique sur le cœur rose pour supprimer des favoris.</p> <p>Alors, le cœur se colore en blanc et l'anime se supprime de la zone des favoris de la page d'accueil.</p>
Résultats obtenus	Le cœur c'est coloré en blanc et l'anime c'est correctement supprimé de la zone des favoris de la page d'accueil.
Statut	✓ OK

Nom	7.1 Affichage du profile de l'utilisateur connecté
User Story	S7 : Affichage du profile
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je clique sur <i>Profile</i> dans la barre de navigation.</p> <p>Alors, la page de profile de l'utilisateur connecté s'affiche avec ses statistiques et ses favoris.</p>
Résultats obtenus	La page de profile de l'utilisateur connecté s'affiche correctement et les statistiques ainsi que les favoris sont les siens.
Statut	✓ OK

Nom	8.1 Affichage des listes de l'utilisateur connecté
User Story	S8 : Affichage des listes
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je clique sur <i>Listes</i> dans la barre de navigation.</p> <p>Alors, la page contenant toutes les listes de l'utilisateur connecté s'affiche ainsi que les animes contenu dans ces listes.</p>
Résultats obtenus	La page contenant les listes de l'utilisateur connecté c'est correctement affiché et les animes sont correctement affiché aussi.
Statut	✓ OK

Nom	9.1 Créer une liste
User Story	S9 : Gestion des listes
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur la page des listes et que j'écris "Ma nouvelle liste" dans le champs de texte <i>Nouvelle liste</i> et que j'appuie sur <input type="button" value="Enter"/>.</p> <p>Alors, la liste apparaîtra en bas des listes déjà présentes avec une  à côté.</p>
Résultats obtenus	La liste à bien été ajoutée en base des listes déjà présente.
Statut	✓ OK

Nom	9.2 Supprimer une liste
User Story	S9 : Gestion des listes
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur la page des listes et que je clique sur  d'une liste présente.</p> <p>Alors, la liste ne sera plus présente dans les listes présentes.</p>
Résultats obtenus	La liste à bien été supprimer et n'est plus présente dans les listes déjà existante.
Statut	✓ OK

Nom	9.3 Renommer une liste
User Story	S9 : Gestion des listes
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je double-clique sur le nom de la liste, je peux renommer la liste et valider en appuyant sur <input type="text" value="Entrée"/>.</p> <p>Alors, le nom de la liste est changé.</p>
Résultats obtenus	Le nom de la listes est bien mis à jour.
Statut	✓ OK

Nom	10.1 Affichage des favoris sur l'accueil
User Story	S10 : Affichage des favoris
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur la page d'accueil.</p> <p>Alors, mes favoris sont présent sur la page.</p>
Résultats obtenus	Mes favoris sont bien affiché.
Statut	✓ OK

Nom	10.2 Affichage des favoris du profile
User Story	S10 : Affichage des favoris
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur ma page de profile.</p> <p>Alors, mes favoris sont présent sur la page.</p>
Résultats obtenus	Mes favoris sont bien affiché.
Statut	✓ OK

Nom	11.1 Organisation des favoris
User Story	S11 : Organisation des favoris
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur la page des favoris et je clique sur le bouton <i>Réorganiser les favoris</i> , je peux glisser déposer les animes dans l'ordre que je veux. Je clique sur le bouton <i>Sauvegarder</i> pour enregistrer l'ordre.</p> <p>Alors, mes favoris sont enregistrer dans l'ordre voulu.</p>
Résultats obtenus	Mes favoris ont bien été réorganisé.
Statut	✓ OK

Nom	11.2 suppression des favoris
User Story	S11 : Organisation des favoris
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je suis sur la page des favoris et je clique sur le bouton <i>Réorganiser les favoris</i> , je peux cliquer sur  pour enlever l'anime des favoris.</p> <p>Alors, l'anime ne fait plus parti des favoris.</p>
Résultats obtenus	L'anime est bien retiré des favoris.
Statut	✓ OK

Nom	12.1 Affichage de la landing page
User Story	S12 : Affichage de la landing page
Situation	<p>Étant donné que je suis un utilisateur non connecté.</p> <p>Quand je suis sur le site.</p> <p>Alors, une page d'accueil s'affiche avec comme possibilité : la visite de la page <i>À propos</i> , se connecter et s'inscrire.</p>
Résultats obtenus	La page d'accueil ainsi que la barre de navigation sont affiché correctement pour un utilisateur non connecté.
Statut	✓ OK

Nom	13.1 Respect du preset Airbnb
User Story	S17 : Vérification syntaxique
Situation	<p>Étant donné que je suis un développeur.</p> <p>Quand j'exécute la commande <code>npm run lint</code> dans le dossier de mon projet.</p> <p>Alors, aucune erreur de syntaxe sur la base du preset Airbnb n'est relevée.</p>
Résultats	<pre>> Animanga@1.0.0 lint /home/cavanat/Documents/programmation/python/Animanga</pre>

obtenus	13.1 Respect du preset Airbnb
Nom	eslint --ignore-pattern node_modules/
Statut	✓ OK

Nom	13.2 Respect des conventions PEP8
User Story	S17 : Vérification syntaxique
Situation	<p>Étant donné que je suis un développeur.</p> <p>Quand j'exécute la commande <code>python3 -m pylint --output-format=colored packages</code> à la racine de mon projet.</p> <p>Alors, aucune erreur de syntaxe sur la base des convention PEP8 n'est relevée.</p>
Résultats obtenus	La note attribuée au code est supérieur à 10/10.
Statut	✓ OK

Nom	14.1 Synchronisation Sqlite3 - MySQL
User Story	S15 : Synchronisation MySQL Sqlite3
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je clique sur l'icône de synchronisation.</p> <p>Alors, la page charge et je suis redirigé vers la page d'accueil.</p>
Résultats obtenus	Les données sont identique entre la base Sqlite3 et MySQL.
Statut	✓ OK

Nom	15.1 Affichage des activités des 24 dernières heures
User Story	S15 : Affichage des activités
Situation	<p>Étant donné que je suis un utilisateur connecté.</p> <p>Quand je met un anime en favoris et dans une liste (changement de statut aussi).</p> <p>Alors, une carte s'affiche sur l'accueil avec le nom de l'anime modifier ainsi que son image, le nom de la liste dans laquelle il a été ajouté, et le temps écoulé depuis la mise à jour.</p>
Résultats obtenus	La carte s'affiche avec les informations correctes.
Statut	✓ OK

Évolution des tests

N° Test	J1 lu.25	J2 ma.26	J3 me.27	J4 je.28	J5 ve.29	J6 ma.2	J7 me.3	J8 je.4	J9 ve.5	J10 lu.8	J11 ma.9
1.1	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.2	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.3	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.4	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.5	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.6	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.7	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.8	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
1.9	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.1	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.2	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.3	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.4	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.5	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
2.6	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗
3.1	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗
4.1	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗	✗
4.2	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗	✗
5.1	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗	✗
6.1	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗	✗
6.2	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗
6.3	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗	✗
6.4	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗	✗
6.5	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗
6.6	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗	✗
7.1	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗	✗
8.1	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗

N° Test	J1 lu.25	J2 ma.26	J3 me.27	J4 je.28	J5 ve.29	J6 sa.30	J7 di.31	J8 je.4	J9 ve.5	J10 sa.6	J11 di.9
9.1	×	×	×	×	×	✓	✓	✓	✓	×	×
9.2	×	×	×	×	×	✓	✓	✓	✓	×	×
10.1	×	×	×	✓	✓	✓	✓	✓	✓	×	×
10.2	×	×	×	✓	✓	✓	✓	✓	✓	×	×
11.1	×	×	×	×	×	✓	✓	✓	✓	×	×
11.3	×	×	×	×	×	✓	✓	✓	✓	×	×
12.1	×	✓	✓	✓	✓	✓	✓	✓	✓	×	×
13.1	×	✓	✓	✓	✓	✓	✓	✓	✓	×	×
13.2	×	✓	✓	✓	✓	✓	✓	✓	✓	×	×
14.1	×	×	×	×	×	×	×	✓	✓	×	×
15.1	×	×	×	×	×	×	✓	✓	✓	×	×

Bibliographie

Voici les différentes ressources techniques consultées lors du développement de mon projet :

- La documentation officielle Python : <https://docs.python.org/3/>
- MDN web docs, recueil très complet concernant le HTML, CSS, et JavaScript, créé par Mozilla : <https://developer.mozilla.org/en-US/>
- Site de questions en lignes pour développeurs de tout les horizons : <https://stackoverflow.com/>
- Le guide de style Airbnb, spécialisé pour la syntaxe JavaScript : <https://github.com/airbnb/javascript>
- Les sources des librairies externes utilisées lors de ce projet :
 - Documentation officielle de Flask : <https://palletsprojects.com/p/flask/>
 - Documentation officielle de Jinja2 : <https://jinja.palletsprojects.com/en/2.11.x/>
 - Documentation officielle de Flask-Login : <https://flask-login.readthedocs.io/en/latest/>
 - Exemple d'utilisation de flask-swagger : <https://pypi.org/project/flask-swagger/>
 - Documentation complète de Mysql Connector/Python : <https://dev.mysql.com/doc/c/connector-python/en/>
 - Explications de ce qu'est Swagger : <https://swagger.io/tools/swagger-ui/>
 - Documentation officielle de JQueryUI : <https://jqueryui.com/>
 - Sources de SwaggerUI, répertoire distant comportant les fichiers utilisés pour l'affichage de mes points d'entrées pour mon API interne : <https://github.com/swagger-api/swagger-ui>

Glossaire

Conclusion

Difficultés majeures rencontrées

Durant tout le développement de mon projet, aucun problème bloquant n'a été rencontré. Voici cependant la liste des soucis les plus majeurs :

- L'outil de fusion de PDF que j'utilisais - **pdfunite** - ne prenait pas en charge les titres lors de la fusion de plusieurs PDF. En effet, si un PDF seul comportait des titres, après la fusion ces derniers étaient considérés comme simple texte.

☐ J'ai pu corriger ce soucis en changeant de librairie de fusion de document PDF. La recherche de nouvelle librairie n'était pas compliquée du tout car il existe un nombre élevé de librairies permettant de fusionner des PDF, dont **pdftk** et le didacticiel disponible à **cette adresse**.

- Lors de la synchronisation, j'utilise des timestamps pour savoir quels enregistrements ont été modifiés. Cependant, j'utilisais un format différent entre ma base SQLite3 et MySQL. Dans SQLite3, le format utilisé était `%Y-%m-%d %H:%M:%f` ce qui donne `2020-06-08 09:08:32.276`. Les millisecondes sont présentes avec ce format. Or, le format utilisé par défaut dans MySQL est `%Y-%m-%d %H:%M:%S` ce qui donne `2020-06-08 09:08:32`. Cette différence de format faisait que lorsqu'un timestamp SQLite3 dont les millisecondes sont plus grandes que `.500`, ce timestamp est arrondi vers le haut et donc mes timestamps sont différents.

☐ Le soucis n'était de loin pas compliqué à régler mais j'ai mis un certain moment afin de trouver ce qui causait le soucis d'enregistrement différent entre SQLite3 et MySQL. Une fois la cause trouvée, j'ai simplement fait en sorte que la date que j'insérerais dans SQLite3 ne comportait pas les millisecondes et tout est rentré dans l'ordre.

Améliorations possibles

Étant donné la courte période mise à disposition, il est clair que des améliorations sont possibles sur les fonctionnalités existantes. Voici un aperçu de ce qui pourrait être amélioré :

- Ajouter la fonctionnalité de pouvoir modifier son profil. Cela comporte le pseudo, email et le mot de passe
- Faire en sorte que l'interface du site soit *responsive design* (qu'il s'adapte sur tout type d'écran). Pour le moment, cette fonctionnalité n'est implémentée qu'à moitié
- Mettre plus de résultats lors d'une recherche. Pour le moment ce n'est que les neuf les plus adéquats par rapport à la chaîne recherchée
- Modifier la fonctionnalité de synchronisation unidirectionnelle entre SQLite3 et MySQL. Pour le moment, l'algorithme utilisé est relativement efficace mais il pourrait être amélioré de cette façon par exemple :

Stocké le timestamp de la dernière synchronisation dans une table, supprimer tout les enregistrements plus présents dans Sqlite3 de MySQL, sélectionner que les enregistrements dont la date est supérieur ou égale à la dernière synchronisation et mettre à jours les enregistrements MySQL et ajouté ceux qui manque.

Cette façon de faire permettrait à l'algorithme d'être beaucoup plus rapide qu'actuellement.

Outre les fonctionnalités existantes, j'ai penser à ces quelques idées durant le développement de l'application :

- Ajouter la fonctionnalité de pouvoir se faire une liste d'amis en cherchant le pseudo de l'utilisateur dans un champs prévu à cet effet et ensuite avoir une page dédié à l'affichage de cette dite liste d'amis afin de pouvoir aller voir le profil de ces derniers.
- Ajouter un système de rôle permettant aux administrateur de pouvoir gérer les animes sans que les utilisateur puisse le faire pour éviter toute fausse manipulation
- Ajouter la fonctionnalité permettant aux utilisateur de mettre une note à un anime et une progression de visionnage (nombre d'épisode regardé)
- Modifier le contenu des activités pour ajouter celles des amis

Bilan personnel

Ce projet m'a énormément plus. Le sujet était parfait pour moi : j'adore réaliser des projet en Python, surtout web, et je suis passionné par les animes. Le fait de pouvoir lier ces deux passions était très agréable. Je trouve très plaisant d'utilisé cette application de part sa simplicité et son contenu fourni. Le fait d'écrire une documentation aussi grosse était une première pour moi et ce fût très enrichissant, en plus de me satisfaire grandement.

Remerciements

J'apporte mes remerciements à :

- M. Pascal Bonvin pour son suivi assidu lors de ce TPI
- M. Nicolas Ettlin pour ses conseils avisé concernant l'utilisation d'eslint et quelque techniques CSS.