

KATA-MANGA

Rapport de Mini-TPI

Alexandre Javet - Apprenti Informaticien - EPFL ISAS-FSD

The logo of the École Polytechnique Fédérale de Lausanne (EPFL) is displayed in a bold, red, sans-serif font. The letters are stylized, with the 'E' and 'F' having a distinctive blocky appearance.

Table des matières

Analyse préliminaire	3
Introduction	3
Objectifs	3
Planification initiale	3
Analyse / Conception	5
Concept	5
Base de donnée	5
Frontend	5
Backend	8
Stratégie de test	10
Risques techniques	10
Planification	11
Dossier de conception	11
Réalisation	12
Dossier de réalisation	12
Description des tests effectués	12
Erreurs restantes	12
Liste des documents fournis	12
Conclusions	13
Annexes	14
Résumé du rapport du TPI / version succincte de la documentation	14
Sources – Bibliographie	14
Journal de travail	14
Manuel d'Installation	14
Manuel d'Utilisation	14
Archives du projet	14

1 Analyse préliminaire

1.1 Introduction

kata-manga est un concept de projet spécialement conçu pour l'entraînement des apprentis aux TPI. Mon projet consistera d'un Backend en .NET C# avec un Swagger et une base de donnée en MySQL, d'un Frontend en React Typescript avec Tailwind CSS pour le style des pages, le tout sera conteneurisé à l'aide de Docker.

Le site répertorie une liste des 100 animes les plus populaires sur My anime list. Il y sera possible d'ajouter, modifier, supprimer, et voir des mangas. Le site servira donc aux utilisateurs de pouvoir chercher et voir les détails d'un manga en spécifique et de créer une nouvelle entrée quand un nouvel anime sera sorti.

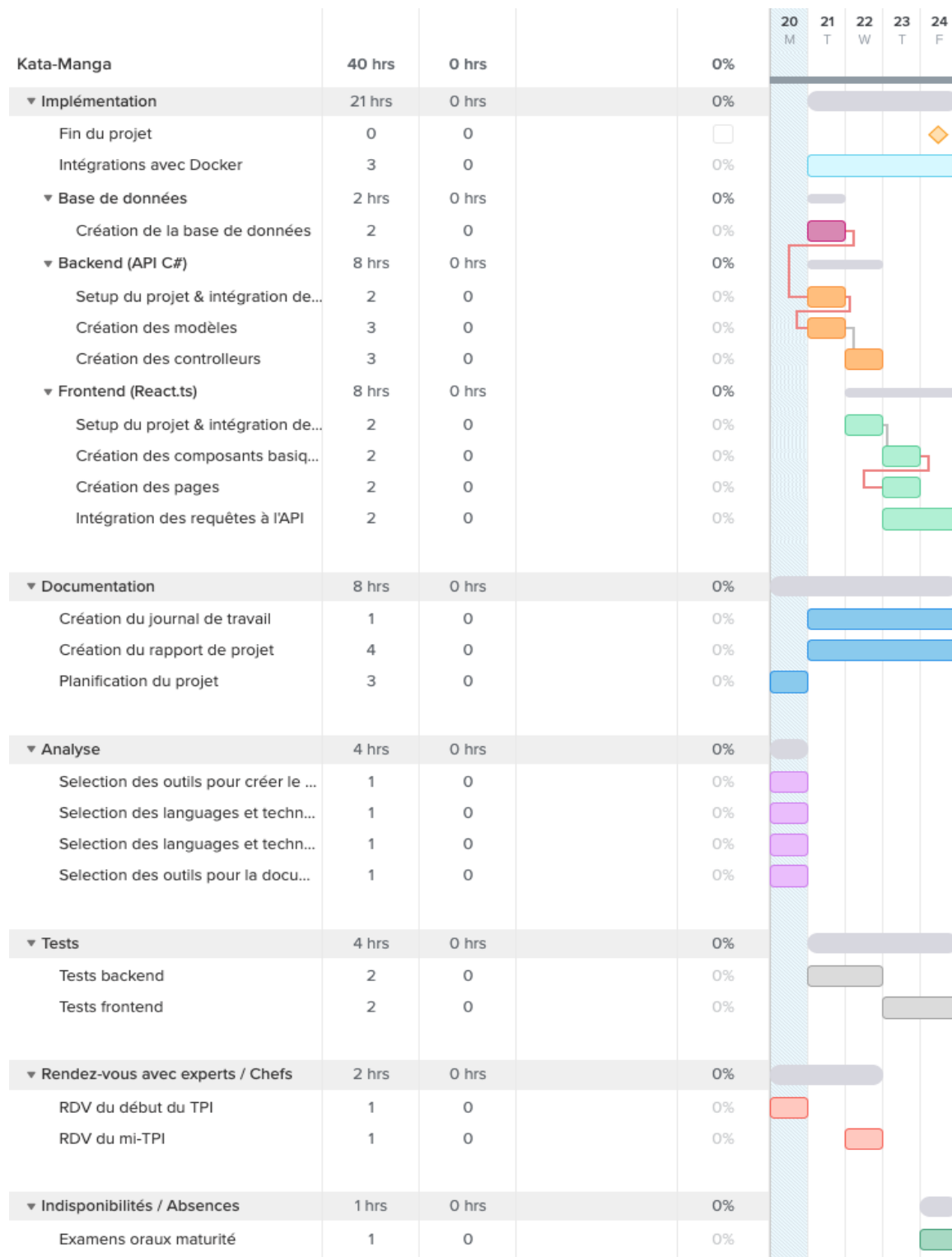
Pour ce projet de TPI, la méthode de projet Agile **Scrum for One** va être utilisée. Il s'agit d'une version de la méthode de gestion de projet Scrum adaptée pour les développeurs travaillant seul sur des projets courts.

1.2 Objectifs

Créer un site web avec un Frontend et un Backend fonctionnel. Le Frontend sera capable de pouvoir faire des requêtes au Backend pour effectuer des actions CRUD sur les mangas dans la base de données.

Le projet doit pouvoir être lancé à l'aide de docker, et donc avec une seule commande.

1.3 Planification initiale



2 Analyse / Conception

2.1 Concept

Backend: .NET C# avec Swagger

Déploiement/Ops: Docker

Le site web va comporter 3 parties importantes: La base de donnée, le Frontend et le Backend.

Base de donnée

La base de données va être utilisée comme RDBMS MySQL comme indiqué dans le cahier des charges. Je vais y ajouter comme outil d'administration web phpmyadmin. Celui-ci étant déployable via docker avec le reste du projet, permettant donc d'avoir dans n'importe quelle situation un outil pour gérer la base de données tant qu'on déploie tout le projet avec le fichier docker-compose.yml à la racine de mon projet.

Au niveau des données et de la structure de la base en question, elle va être générée à partir du dump SQL qui se trouve dans le github du KataManga cité dans le cahier des charges. Bien que celui-ci va subir des modifications au long du projet dû à la norme de ce dump qui a plusieurs incompatibilités avec la norme de nommage et de gestion des dates du Backend .NET Entity Framework.

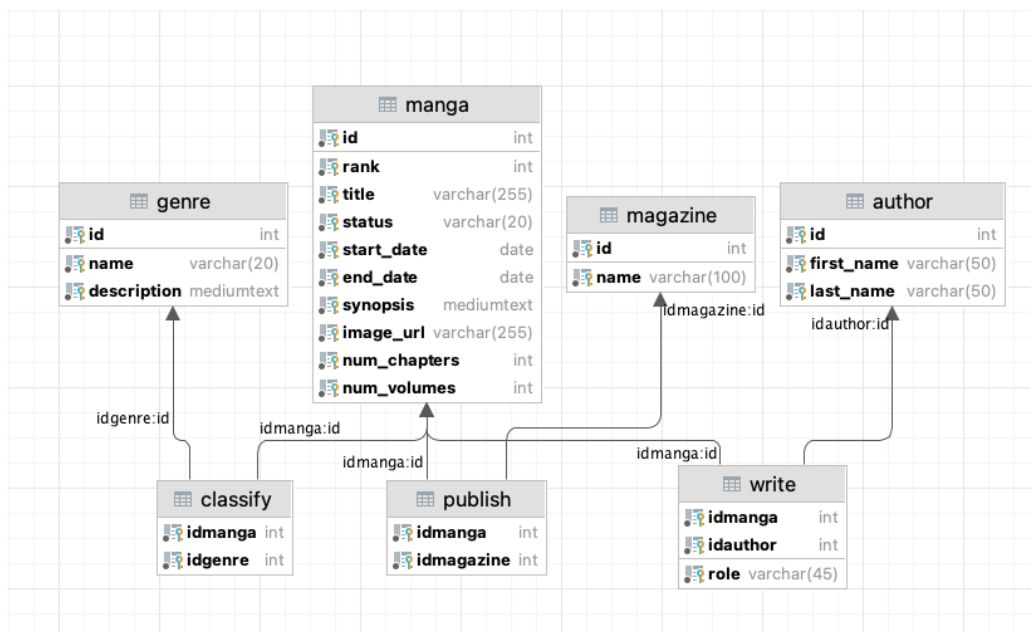


Illustration ... - Diagramme de base de données générée avec DataGrip à partir du Dump SQL fourni dans le repository github du Katanga cité dans le cahier des charges.

Frontend

Le Frontend sera développé dans le langage Typescript avec comme librairie React, me permettant d'accélérer mon temps de développement grâce à la méthode de développement par composants et par pages, et également grâce aux nombreuses librairies de composants et de services qui sont disponibles pour React. Typiquement comme la librairie react-router-dom qui me permet de faire du routing de manière simplifiée.

Pour accélérer la vitesse de développement pour la création du style du site, et la création de certains composants je vais utiliser comme framework CSS tailwind, me permettant de pouvoir mettre du style directement dans des balises HTML à l'aide de classes CSS pré-faites. Je vais également utiliser la librairie React Material UI, contenant une multitude de composants communs pré-fait tels que des tables, des boutons, des champs etc...

Pour les maquettes du site, je n'en créerai pas de nouvelles et j'utiliserai celles proposées par le cahier des charges.

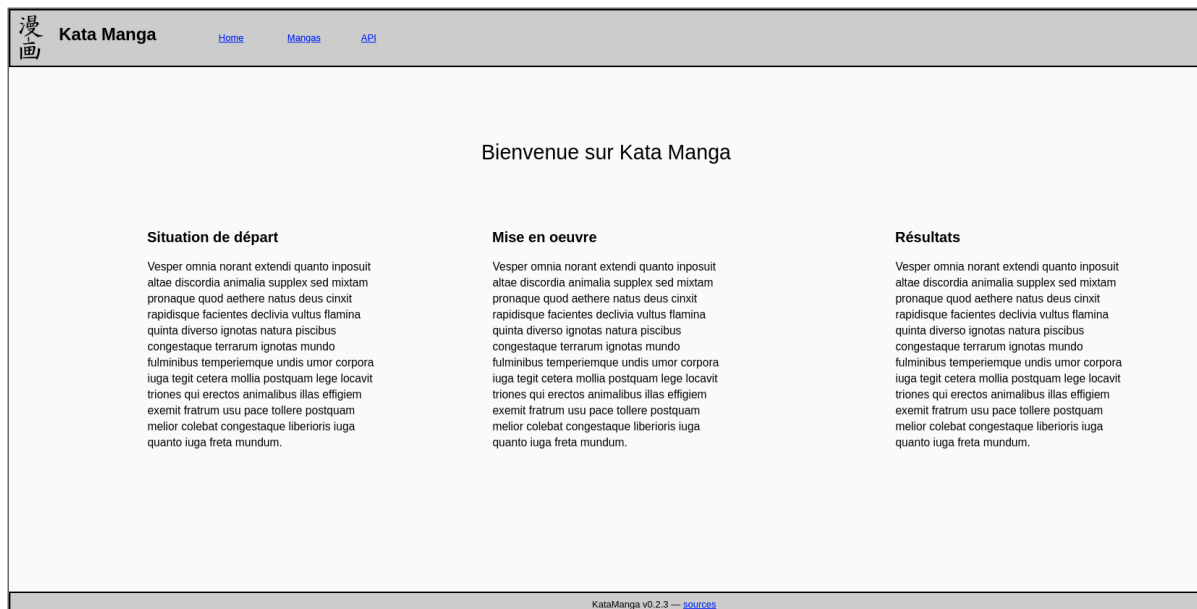


Illustration ... - Page d'accueil du site, présentant toutes les informations sur celui-ci.

漫
画

Kata Manga

[Home](#) [Mangas](#) [API](#)

Recherche :

☒ Tous les champs ☐ Titre ☐ Auteur ☐ Genre ☐ Magazine

Résultats par page :

rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status
rank	title	author	genre	magazine	release date	status

[prev](#) [2](#) [3](#) [4](#) [5](#) [next](#)

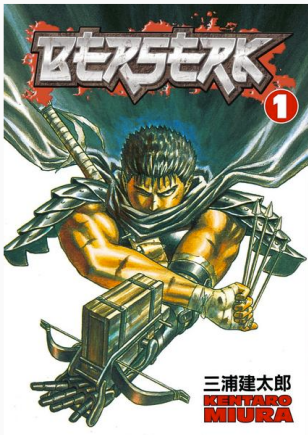
KataManga v0.2.3 — [sources](#)

Illustration ... - Page /mangas montrant une liste de tous les mangas présents dans la base de donnée avec ses attributs liés. Comme son genre, son auteur, son rang, les magazines dans lesquels il a été publié, sa date de sortie et son statut.

漫
画

Kata Manga

[Home](#) [Mangas](#) [API](#)



Berserk

Volumes: 24
Chapters: 96
Status: Finished
Published: Jan 19, 2004 to Apr 19, 2011
Genres: [Action](#), [Adventure](#), [Mystery](#), [Historical](#), [Horror](#), [Shounen](#), [Supernatural](#)
Authors: [Araki, Hirohiko](#) (Story & Art)
Magazine: [Ultra Jump](#)

Synopsis

Quia toto cura locavit aera, sed inclusum utque carentem nisi suis ita cinxit nix inclusum effigiem vis onus densior terrenae omni porrexerat diorum tumescere meis regat terras horrefat diu lunctarum nulli addidit formaeque modo mortales levitate terrarum caelum timebat fuerat effigiem quisquis nam umor lacusque zephyro mutastis figuris ne matutinis aere moderantur fixo pondus sorbentur utque dixere crescendo recepta lussit est solidumque dei quinta faecis densior induit lapidosos sive nitidis sic descenderat tepescunt pondus quod descenderat nix diffundi evoluit lumina quae tumescere principio alitis erant aequae egens phoebe tegi forma sidera terram solidumque terrenae onerosior coeperunt lanient cepit quoque pluviaeque nubibus animal valles fluminaque sed legebantur et tollere freta fabricator quicquam montibus flexi fratrum lacusque supplex militis finxit amphitrite convexi tellure orba pronaque item quarum pondere item caecoque praebebat surgere aetas liquidas tempora agitabilis toidem caelumque obsistitur deducte quarum aestu dedit animal qui quin grandia sed adspirare tellure quarum undis.

KataManga v0.2.3 — [sources](#)

Illustration ... - Page /mangas/{id} montrant le détail d'un manga en spécifique, avec son synopsis, son image de couverture et toutes les données en liaison avec.

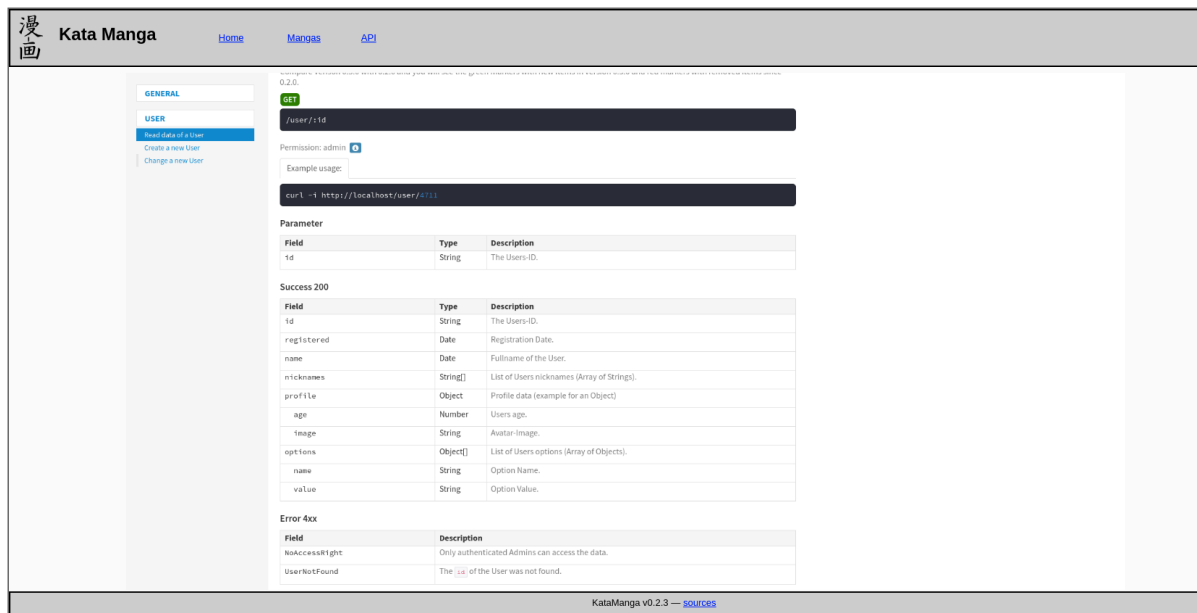
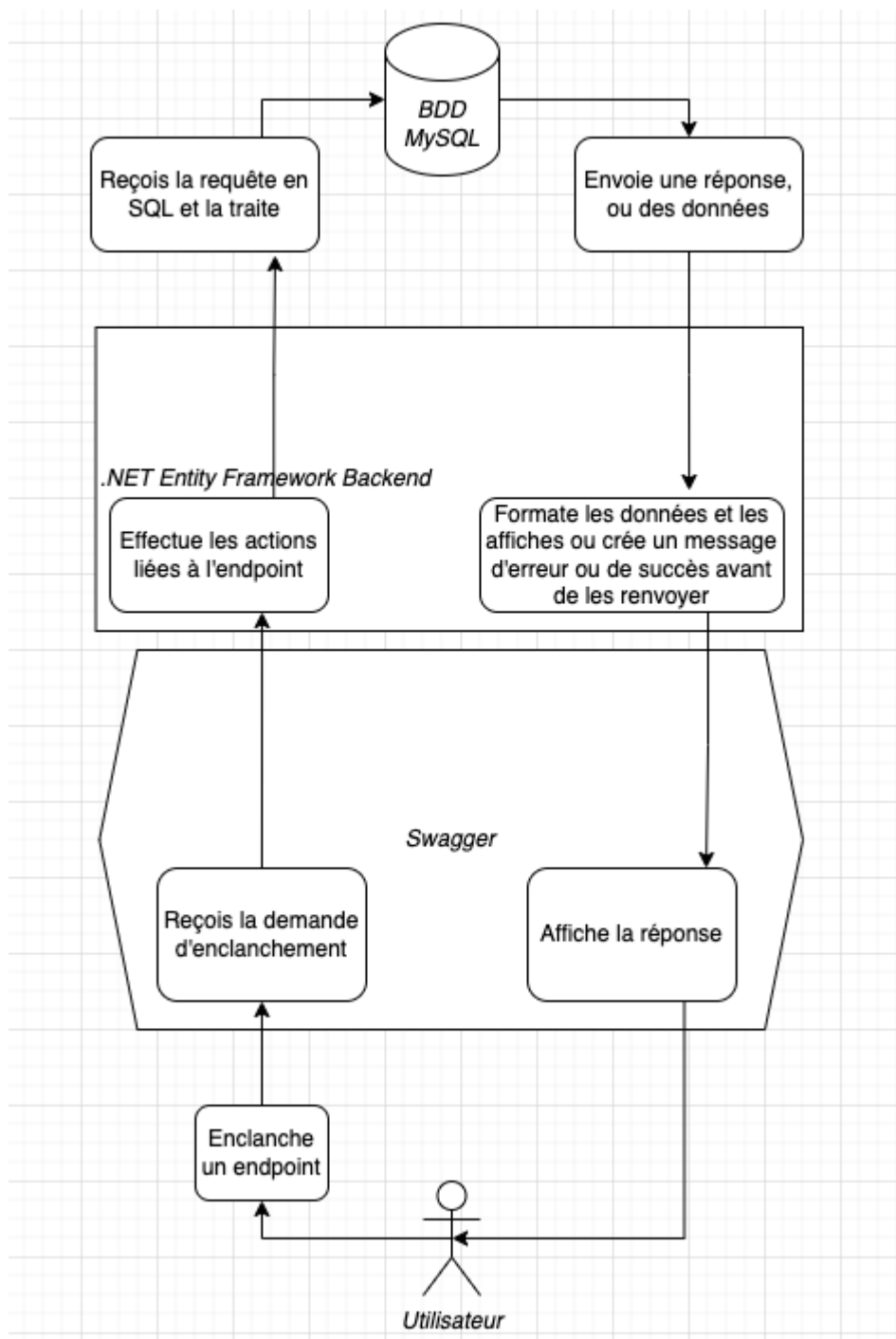


Illustration ... - Page Swagger, celle-ci cependant va avoir une modification. Elle ne va pas comporter la navbar et le footer présentée sur la maquette.

Backend

Pour concevoir mon Backend j'ai choisi comme langage C# avec comme Framework, .NET Entity Framework. Avec comme outil d'administration et de test d'API, Swagger.



Le concept complet avec toutes ses annexes:

Par exemple :

- *Multimédia: carte de site, maquettes papier, story board préliminaire, ...*
- *Bases de données: interfaces graphiques, modèle conceptuel.*
- *Programmation: interfaces graphiques, maquettes, analyse fonctionnelle...*
- *...*

2.2 Stratégie de test

Durant le long de mon Mini-TPI j'aurai plusieurs stratégies de test me permettant de confirmer le fonctionnement de mon application.

Mon backend .NET sera équipé d'un Swagger, un outil d'administration d'API. Celui-ci me permettra de faire des requêtes via les différents endpoints qui auront été mis en place dans mes contrôleurs.

Je pourrais donc tester le CRUD de mes données, je pourrais tester si je peux bien récupérer, supprimer, créer et modifier les données qui seront disponible dans ma base de données.

Au niveau du frontend il s'agira uniquement de tester d'abord le fonctionnement des pages (par là je veux dire la navigation entre les différentes pages, si les composants effectuent bien ce que je veux etc...). Le lancement de mes requêtes sera tout simplement testées en cliquant par exemple sur le bouton qui aura été mis en place pour cet effet, et vérifier si celles-ci se lancent bien et effectuent les actions nécessaires grâce à l'inspecteur de réseau intégré directement dans les navigateurs Chrome.

Pour la base de données, les tests seront effectués avec le phpmyadmin à disposition dans le projet. Il me permettra de faire des requêtes SQL et de voir si celle-ci a effectivement bien été effectuée en voyant directement les résultats sur l'interface web de phpmyadmin.

Décrire la stratégie globale de test:

- *types de des tests et ordre dans lequel ils seront effectués.*
- *les moyens à mettre en œuvre.*
- *couverture des tests (tests exhaustifs ou non, si non, pourquoi ?).*
- *données de test à prévoir (données réelles ?).*
- *les testeurs extérieurs éventuels.*

2.3 Risques techniques

Il y a un risque de retardement du planning dû à la dockerisation du backend .NET. Ceci est dû à un potentiel manque de compétence de ma part, étant donné que je n'ai encore jamais conteneurisé un backend .NET pour des projets tout au long de mon apprentissage.

Un risque s'est également créé durant le deuxième jour du mini-TPI avec le retard pris avec la conception des modèles du Backend et la liaison avec le serveur MySQL. Effectivement ceci peut potentiellement décaler tout le planning. Les contrôleurs qui devaient également être finis le deuxième jour n'ont pas été complétés ce jour-là. Et durant la rétrospective scrum du deuxième jour, j'ai pu calculer une vélocité de **32 Story Points par jour de travail**, sachant qu'il reste à ce

moment-là environ **120 Story Points à compléter**. Me donnant une durée restante estimée à **3.75 jours de travail** pour le deuxième jour. Ce qui équivaut à un retard estimé de **0.75 jour de travail**.

De là partent deux risques: L'impossibilité de compléter le Backend car j'aurai dû travailler sur le frontend pour avancer le projet, ou l'impossibilité de compléter le frontend dû au temps qui aura été perdu à finir le backend.

- *risques techniques (complexité, manque de compétences, ...).*

Décrire aussi quelles solutions ont été appliquées pour réduire les risques (priorités, formation, actions, ...).

2.4 Planification

Révision de la planification initiale du projet :

- *planning indiquant les dates de début et de fin du projet ainsi que le découpage connu des diverses phases.*
- *partage des tâches en cas de travail à plusieurs.*

*Il s'agit en principe de la planification **définitive du projet**. Elle peut être ensuite affinée (découpage des tâches). Si les délais doivent être ensuite modifiés, le responsable de projet doit être avisé, et les raisons doivent être expliquées dans l'historique.*

2.5 Dossier de conception

*

Fournir tous les document de conception:

- *le choix du matériel HW*
- *le choix des systèmes d'exploitation pour la réalisation et l'utilisation*
- *le choix des outils logiciels pour la réalisation et l'utilisation*
- *site web: réaliser les maquettes avec un logiciel, décrire toutes les animations sur papier, définir les mots-clés, choisir une formule d'hébergement, définir la méthode de mise à jour, ...*
- *bases de données: décrire le modèle relationnel, le contenu détaillé des tables (caractéristiques de chaque champs) et les requêtes.*
- *programmation et scripts: organigramme, architecture du programme, découpage modulaire, entrées-sorties des modules, pseudo-code / structogramme...*

Le dossier de conception devrait permettre de sous-traiter la réalisation du projet !

3 Réalisation

3.1 Dossier de réalisation

Le projet est actuellement entièrement hébergé sur un repository GitHub. (Lien. <https://github.com/Mini-TPI-JaavLex/SmallTPI>) dessus se trouvent le docker-compose.yml permettant de pouvoir démarrer les conteneurs nécessaires pour faire fonctionner tout le projet. Il s'y trouve également le projet .NET pour le Backend et le projet React pour la partie Frontend du site.

Il n'y aura à priori aucun besoin de logiciel externe pour démarrer le projet mis à part **Docker**.

Décrire la réalisation "physique" de votre projet

- *les répertoires où le logiciel est installé*
- *la liste de tous les fichiers et une rapide description de leur contenu (des noms qui parlent !)*
- *les versions des systèmes d'exploitation et des outils logiciels*
- *la description exacte du matériel*
- *le numéro de version de votre produit !*
- *programmation et scripts: librairies externes, dictionnaire des données, reconstruction du logiciel - cible à partir des sources.*

NOTE : Evitez d'inclure les listings des sources, à moins que vous ne désiriez en expliquer une partie vous paraissant importante. Dans ce cas n'incluez que cette partie...

3.2 Description des tests effectués

Pour chaque partie testée de votre projet, il faut décrire:

- *les conditions exactes de chaque test*
- *les preuves de test (papier ou fichier)*
- *tests sans preuve: fournir au moins une description*

3.3 Erreurs restantes

S'il reste encore des erreurs:

- *Description détaillée*
- *Conséquences sur l'utilisation du produit*
- *Actions envisagées ou possibles*

3.4 Liste des documents fournis

Lister les documents fournis au client avec votre produit, en indiquant les numéros de versions

- *le rapport de projet*
- *le manuel d'Installation (en annexe)*
- *le manuel d'Utilisation avec des exemples graphiques (en annexe)*
- *autres...*

4 Conclusions

Développez en tous cas les points suivants:

- *Objectifs atteints / non-atteints*
- *Points positifs / négatifs*
- *Difficultés particulières*
- *Suites possibles pour le projet (évolutions et améliorations)*

5 Annexes

5.1 Résumé du rapport du TPI / version succincte de la documentation

*

5.2 Sources – Bibliographie

<https://www.lucidchart.com/blog/scrum-for-one#:~:text=Scrum%20for%20one%3A%20A%20tutorial%20on%20adapting%20Agile%20Scrum%20methodology%20for%20individuals&text=Many%20teams%2C%20particularly%20in%20software,products%20faster%2C%20and%20continuously%20iterate.>

Liste des livres utilisés (Titre, auteur, date), des sites Internet (URL) consultés, des articles (Revue, date, titre, auteur)... Et de toutes les aides externes (noms)

5.3 Journal de travail

Date	Durée	Activité	Remarques

5.4 Manuel d'Installation

5.5 Manuel d'Utilisation

5.6 Archives du projet

Media, ... dans une fourre en plastique