

**Dossier de conception**

>> Diginamic.fr

|  |  |  |  |
| --- | --- | --- | --- |
| **Révision** | **Rédacteurs** | **Date** | **Objet** |
| 1 | 1. CICCOLI, MONTCHO G, VINCKIER N | 02/12/2024 | Création du document – Version initiale |

# Introduction

## Objet du document

Ce document a pour objectif de présenter l’essentiel des questions techniques liées à la mise en place de l’API SIGPC.

Ce document présente :

* L’architecture logicielle de l’API
* Le diagramme de classes
* Le diagramme entités relations
* Le découpage en couches
* Les règles de développement et la qualité de code

# Sommaire

### Table des matières

[1 Introduction 2](#_Toc183176990)

[1.1 Objet du document 2](#_Toc183176991)

[2 Sommaire 3](#_Toc183176992)

[2.1.1 Table des matières 3](#_Toc183176993)

[3 Architecture logicielle 4](#_Toc183176994)

[3.1 Produits et versions 4](#_Toc183176995)

[3.1.1 Langages, frameworks et librairies spécifiques 4](#_Toc183176996)

[3.1.2 Serveur de base de données 4](#_Toc183176997)

[4 Focus Technique 5](#_Toc183176998)

[4.1 Diagramme de classes 5](#_Toc183176999)

[4.2 Diagramme entités relations 5](#_Toc183177000)

[4.3 Découpage en couches 5](#_Toc183177001)

[4.4 Règles de développement et qualité de code 5](#_Toc183177002)

# Architecture logicielle

## Produits et versions

### Langages, frameworks et librairies spécifiques

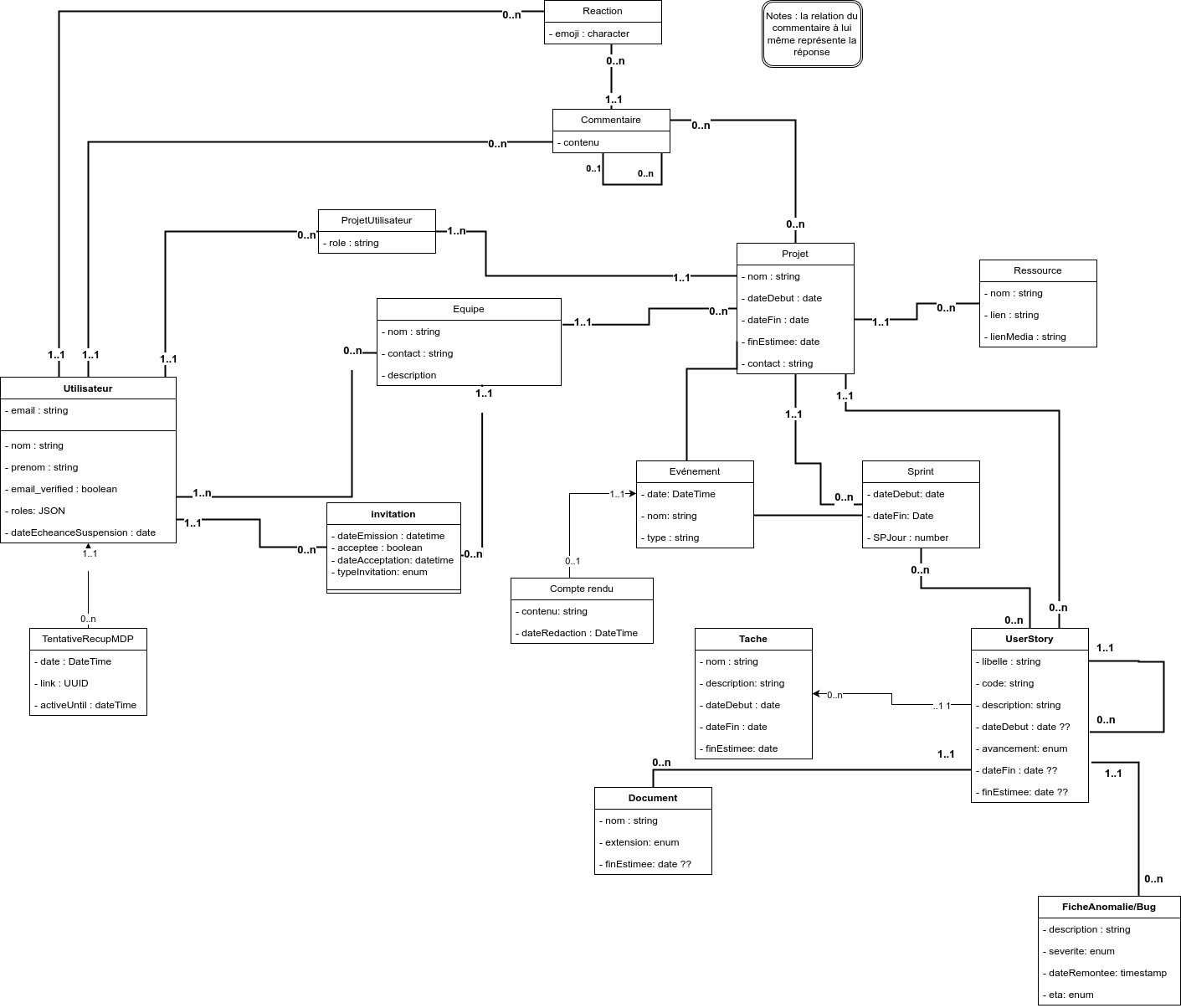
|  |  |
| --- | --- |
| **Nom** | **Version** |
| Langage Java | 21 |
| Spring boot | 3.3.3 |
| Spring data JPA | 3.3.3 |
| Spring web | 6.1.12 |
| Java JWT | 4.4.0 |
| OpenAPI | 3.1.0 |
| Spring HATEOAS | 2.4.0 |

### Serveur de base de données

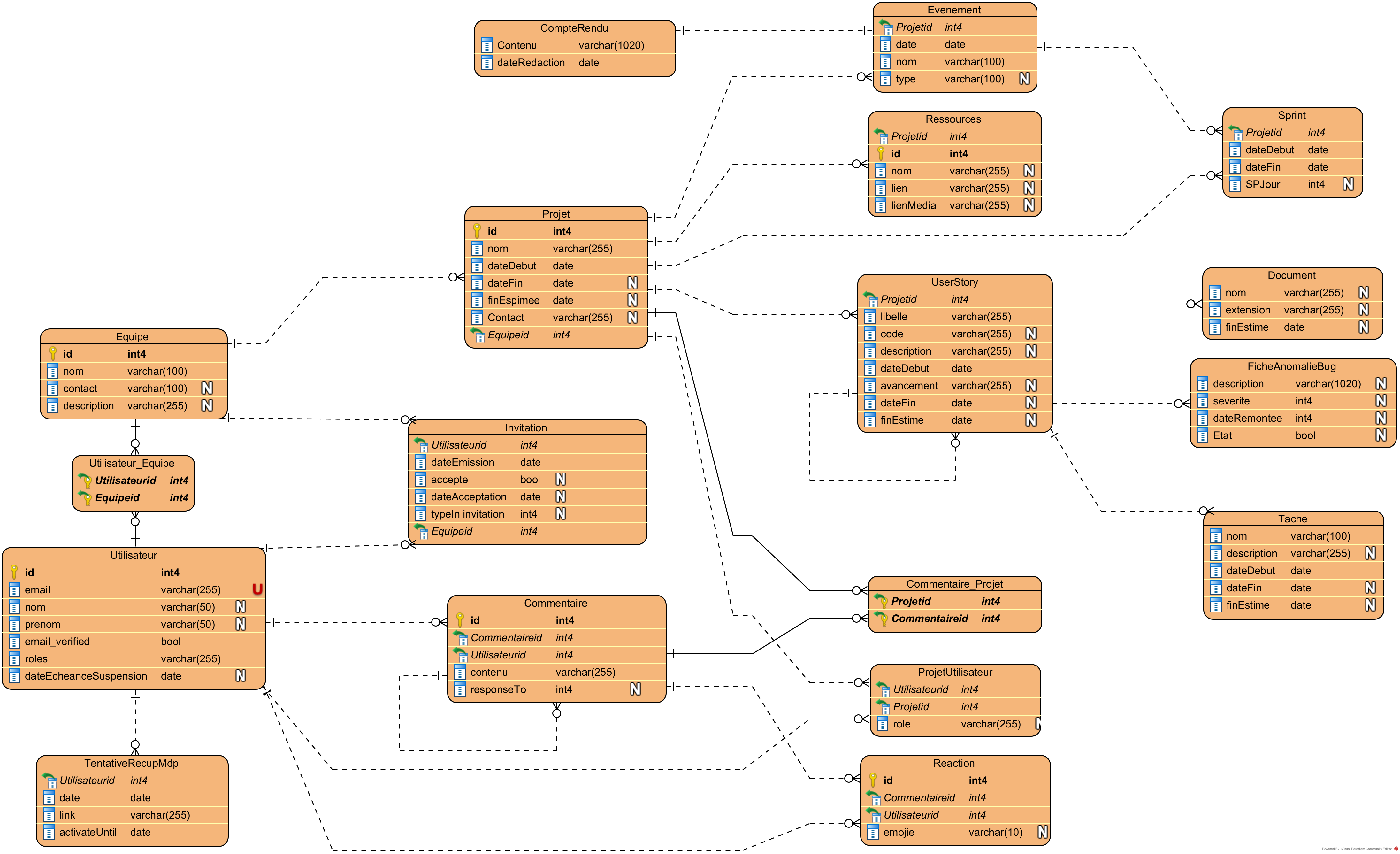
|  |  |
| --- | --- |
| **Nom** | **Version** |
| MariaDB / MySQL | 11.16.2 / 8.0.40 |
|  |  |

# Focus Technique

## Diagramme de classes



## Diagramme entités relations



## Découpage en couches

Les couches seront organisées selon le principe d’architecture hexagonale

- Couche controller : Correspond aux méthodes exposant les points d’entrée de l’API de l’application Spring Boot.

- Couche service : Correspond aux couches orchestrant la validation, la transformation et la transmission des données entre l’API de l’application et sa base de données.

- Couche validateur : Correspond aux classes s’assurant de la validité des données reçues de l’API avant transmission aux repositories. À voir en pratique.

- Couche repository : Correspond aux couches faisant les appels à la base de données. Ne contiennent aucune logique métier.

- Couche entité : Correspond à la couche représentant la transcription des tables de la base de données en objets Java.

- Couche DTO : Correspond à la couche représentant la transcription des données reçues et transmises via l’API de JSON à des objets Java

- Couche mapper : Correspond à la couche se chargeant de faire la transcription des DTOs aux entités et inversement.

- Couche config : Couche contenant la sécurité, le lien aux ressources externes comme les variables d’environnement, et la gestion du cycle de vie de l’application

## Liste des endpoints

Liste des endpoints de l’API :

Tous les endpoints commencent par **/api/v1**

|  |  |
| --- | --- |
| Description | Endpoint |
| Permet de créer un compte | POST /auth/signup |
| Permet de vérifier son adresse email | GET /auth/verify/:uuid |
| Permet de s'authentifier | POST /auth/login |
| Permet d'envoyer une demande de récupération de mdp | POST /auth/password/renew |
| Permet de changer son mot de passe | POST /auth/password/change/:uuid |
| Permet de créer une équipe | POST /group |
| Permet de lancer une invitation pour rejoindre un groupe | POST /group/:group\_id/invite |
| Permet de demander à rejoindre un groupe | POST /group/:group\_id/join |
| Permet d'accepter / refuser un utilisateur dans son groupe | PATCH /group/:group\_id/join/:accepted  *accepted = true ou false* |
| Permet d'accepter / refuser une invitation à un groupe | PATCH /group/:group\_id /invite/member/:accepted  *accepted = true or false* |
| Permet de sortir d'une équipe | PATCH /group/:group\_id/leave |
| Permet de changer les informations d'une équipe | PUT /group/:group\_id |
| Permet de supprimer un membre de l'équipe | DELETE /group/member/:member\_id |
| Permet de changer le rôle d'un membre de l'équipe | PATCH /group/:group\_id/member/:mb\_id/role |
| Permet de créer un projet | POST /project/:group\_id |
| Permet de rajouter un lien à un projet | POST /project/:project\_id/link |
| Permet de lier un document à un projet | POST /project/:project\_id/file |
| Permet de supprimer un document attaché à un projet | DELETE /project/file/:file\_id |
| Permet d'ajouter un commentaire à un projet | POST /comment/:project\_id |
| Permet d'ajouter une réponse à un commentaire | POST /comment/reply/:comment\_id |
| Permet de réagir à un commentaire | POST /comment/react/:comment\_id |
| Permet d’ajouter le membre d’une équipe à un projet | POST /project/:project\_id/add-member/:user\_id |
| Permet de créer une User Story | POST /user-story/:project\_id  *Note: parent\_id dans le body (peut être NULL)* |
| Permet de changer une User Story | PUT /user-story/:user\_story\_id |
| Permet de changer la progression d’une User Story | PATCH /user-story/:user\_story\_id/progression |
| Permet de supprimer une User Story | DELETE /user-story/:user\_story\_id |
| Permet de créer une Fiche de bug | POST /bug-report/:user\_story\_id |
| Permet de modifier une Fiche de bug | PUT /bug-report/:bug\_report\_id |
| Permet de supprimer une Fiche de bug | DELETE /bug-report/:bug\_report\_id |
| Permet de modifier une tâche associée à une user-story | PUT /task/:task\_id |
| Permet de créer une tâche associée à une user story | POST /task/:user\_story\_id |
| Permet de supprimer une tâche associée à une user-story | DELETE /task/:task\_id |
| Permet d’ajouter un sprint au projet | POST /sprint/:project\_id |
| Permet de modifier un sprint du projet | PUT /sprint/:sprint\_id |
| Permet de supprimer un sprint du projet | DELETE /sprint/:sprint\_id |
| Permet d’ajouter une liste de user-stories au sprint | PUT /sprint/backlog/:sprint\_id |
| Permet de supprimer une liste de user-stories au projet | DELETE /project/backlog/:project\_id |
| Permet d’ajouter un événement à un projet | POST /event/projet/:project\_id |
| Permet d’ajouter un événement à un sprint | POST /event/sprint/:sprint\_id |
| Permet de récupérer les éléments du planning | GET /event |
| Permet de récupérer le backlog du sprint | GET /sprint/backlog |
| Permet de récupérer le backlog du projet | GET /project/backlog |
| Récupérer la liste des taches du sprint | GET /sprint/tasks |
| Récupérer la liste des taches du projet | GET /project/tasks |
| Récupérer la liste des user stories d’un projet | GET /user-story/:project\_id |
| Permet de récupérer la liste des projets pour une équipe | GET /project/:group\_id |
| Permet de récupérer la liste des utilisateurs d’un projet donné | GET /project/users/:group\_id |
| Permet de récupérer la liste des utilisateurs d’un groupe donné | GET /user/:group\_id |
|  |  |

## Règles de développement et qualité de code

Règles de développement :

* Javadoc à 100% sur les classes des controllers, services et validateurs.
* Javadoc sur toutes les méthodes de repository utilisant une requête native
* Respect des normes de nommage

Toute fonctionnalité doit être livrée avec des requêtes Postman / Insomnia / Curl

Il y aura des tests unitaires à développer pour tester le code des classes des validateurs et des classes de service non triviales.

On considère non triviale une classe de service contenant des conditions ou des calculs. Si une classe de service se contente de faire des appels dans l’ordre à d’autres parties du code, elle est considérée triviale.

Si la validation des données en entrée se fait dans la couche service, les méthodes doivent obligatoirement être testées.

Il y aura également des tests d’intégration à développer sur chaque point de l’API n’étant pas un GET.

Cependant, le temps de développement étant limité, on définit les priorités suivantes:

- Développement des fonctionnalités avec la doc et les requêtes permettant de tester manuellement l’API

- Développement des tests unitaires sur la validation des données en entrée

- Développement des tests d’intégration