05/03/2024

Tiago Rodrigues Sousa, Evin Paramanathan

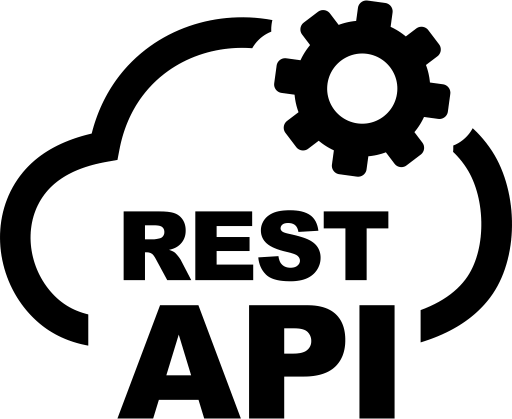
ETML – Vennes

Chef de projet : Antoine Mveng

Salle : B11

Rapport de projet

P\_Web\_C295



Rapport de projet

Table des matières

[Introduction 2](#_Toc160907674)

[Analyse 2](#_Toc160907675)

[1. Planification 2](#_Toc160907676)

[2. Verbe http 2](#_Toc160907677)

[Routes : 2](#_Toc160907678)

[3. Base de données 3](#_Toc160907679)

[4. Structure du code 4](#_Toc160907680)

[5. Schéma de l’architecture 5](#_Toc160907681)

[Réalisation 5](#_Toc160907682)

[1. Authentification et la gestion des rôles 5](#_Toc160907683)

[2. Sécurité 5](#_Toc160907684)

[3. Fonctionnalité technique 5](#_Toc160907685)

[b) Swagger 5](#_Toc160907686)

[c) Docker 5](#_Toc160907687)

[Test 5](#_Toc160907688)

[ChatGPT 5](#_Toc160907689)

[Conclusion 6](#_Toc160907690)

[1. Gestion du code 6](#_Toc160907691)

[2. Général 6](#_Toc160907692)

[3. Personnelle 6](#_Toc160907693)

[a) Tiago 6](#_Toc160907694)

[b) Evin 6](#_Toc160907695)

[4. Planification du projet 6](#_Toc160907696)

[Webographie 6](#_Toc160907697)

[1. Docker : 6](#_Toc160907698)

# Introduction

L'objectif de ce projet est de mettre en pratique le module ICT C295, qui consiste à réaliser le backend pour des applications. Pour ce faire, nous devons mettre en place une API qui permet de partager notre passion pour la lecture. Nous travaillons en binôme pour faire ce projet.

Pour réaliser ce projet, nous disposons d'un PC, de VS Code, d'un accès à Internet, de supports de cours, ainsi que de technologies telles que Node.js, npm, Express.js, Sequelize, MySQL, Swagger et Insomnia. Nous avons à notre disposition 24 périodes pour faire ce projet.

Avant d’aller dans les détails regardons juste qu’est-ce que c’est une API.

API est l’abréviation d’application programming interface ou interface de programmation d’application en français, une api est une interface logicielle qui permet de relier d’un service a un autre ou d’un logiciel a un autre. Cela permet d’échanger des données et des fonctionnalités

# Analyse

## Planification

La planification des taches à réaliser pour le projet a été fait sur Trello, ça permet de créer un tableau et de l’adapter pour faire un suivit du projet plus personnalisé.

<https://trello.com/invite/b/CKSDCYuq/ATTI9eb058082f456ec9f8f1b74df1999ac35786C719/pweb295>

## Verbe http

|  |  |
| --- | --- |
| Verb | Action |
| GET | Recherche |
| POST | Créer |
| PUT | Mettre à jours |
| DELETE | Supprimer |

## Routes :

Livre :

|  |  |  |  |
| --- | --- | --- | --- |
| Nom du fichier | Verbe | URI | Action |
| addLivre | POST | http://localhost:3000/api/livres | Permet d’ajouter un livre |
| deleteLivre | DELETE | http://localhost:3000/api/livres/3 | Permet de supprimer un livre |
| getAllLivresCategorie | GET | http://localhost:3000/api/categories/1/livres | Permet de rechercher un le livre spécifique d’une catégorie |
| getLivre | GET | http://localhost:3000/api/livres | Permet de rechercher un livre spécifique |
| getLivreId | GET | http://localhost:3000/api/livres/2 | Permet de rechercher un livre spécifique grâce à son IP |
| updateLivre | PUT | http://localhost:3000/api/livres/2 | Permet de mettre à jour un livre |

Catégorie :

|  |  |  |  |
| --- | --- | --- | --- |
| **addCategorie** | POST | http://localhost:3000/api/categories | Permet d’ajouter une catégorie |
| **getCategorie** | GET | http://localhost:3000/api/categories | Permet de rechercher une catégorie |
| **getCategorieId** | GET | http://localhost:3000/api/categories/1 | Permet de rechercher une catégorie à partir d’un id |

Commentaire :

|  |  |  |  |
| --- | --- | --- | --- |
| **addCommentaires** | POST | http://localhost:3000/api/commentaires | Permet d’ajout un commentaire |
| **getCommentaire** | GET | http://localhost:3000/api/commentaires | Permet de rechercher des commentaire |

Login :

|  |  |  |  |
| --- | --- | --- | --- |
| **Login** | POST | http://localhost:3000/api/login |  |

## Base de données

Pour faire la base de données nous avons décider de faire 6 tableaux, comme on peut le voir dans l’image ci-dessous.

Premièrement Il y a la table principale ouvrage, celle-ci et relier à 5 tableaux. Il y a la table catégorie qu’on a décidait de pas mettre directement dans ouvrage, car si un livre à plusieurs catégories ça aller posait un problème pour les recherches.

Deuxièmement, il y a les deux tables qui sont écrivain et Editeur, pour chaque Ecrivain et chaque Editeur.

Ensuite il y a la table utilisateur

Une image contenant texte, capture d’écran, diagramme, Post-it

Description générée automatiquement

Une image contenant texte, capture d’écran, diagramme, ligne

Description générée automatiquement

## Structure du code

Le projet (partie code) est organisé de cette façon : dans le dossier API qui est la racine de l’API il y a un gitignore, les packages json et un dossier src, à l’intérieur de celui-ci il y a un fichier app.mjs et 4 dossiers.

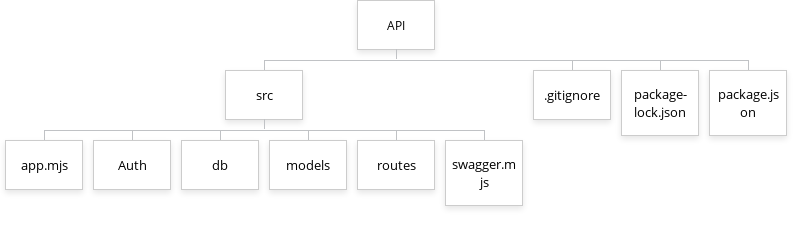
Premièrement il y a le dossier auth contient deux fichiers pour faire le système d’authentification auth.mjs et private\_key.mjs.

Ensuite il y a le dossier db qui contient aussi deux fichiers, mocks-livre.mjs et sequelize.mjs

L’avant dernière dossier est le dossier models qui contient 6 fichiers, chaque fichier est une table de la base données : donc il y a les fichiers suivant t\_categorie.mjs, t\_commentaire.mjs, t \_ecrivain.mjs, t\_editeur.mjs, t\_ouvrage.mjs et t\_utilisateur.

Pour finir il y a le dossier routes qui contient le plus de fichier. Elle contient toutes les fonctionnalités de l’API donc il y a 12 fichiers dans ce dossier voici la liste de fichiers :

* addCategorie.mjs
* addCommentaires.mjs
* addLivre.mjs
* deleteLivre.mjs
* getAllLivreCategorie.mjs
* getCategorie.mjs
* getCategorieId.mjs
* getCommantaire.mjs
* getLivre.mjs
* getLivre.mjs
* getLivreId.mjs
* helper.mjs
* login.mjs
* updateLivre.mjs



## Schéma de l’architecture

# Réalisation

## Authentification et la gestion des rôles

Pour l’instant, il y n’y a qu’un champ utiRole dans la table utilisateur ou il y admin et user. Ils ne sont pour l’instant pas utilisé pour faire la gestion des rôles. Cela n’a pas été fait par manque de temps.

## Sécurité

Un système de token a été mis en place. Elle est obligatoire pour pouvoir utiliser l’api. Nous avons également mis un hachage pour le mot de passe qui éviter que le mot de passe soit écrit en dur dans la base de données. Ce sont les seules sécurités qui ont été mises en place par manque de temps.

# Test

# ChatGPT

Pour la partie code chatGPT nous a servis à comprendre des détails qu’on ne savait pas ou qu’on ne comprenait pas. ChatGPT nous a aussi servit à créer des données factices pour les insérer dans la base de données.

Et du côté du rapport ChatGPT nous a seulement aider à corriger des fautes d’orthographe et de grammaire.

# Conclusion

## Gestion du code

Nous avons utilisé GitHub comme gestionnaire de versionning. Il y avait 4 branches :

* main : rassemble toutes les branches
* MiseEnPlaceAPI : c’est là où est stocké le code de l’api et insomnia
* Database : il y a le MCD/MLD/MPD
* Documentation : stocke la documentation

## Général

Le projet est plutôt abouti toutes les fonctionnalités créer dans le dossier marche, mais le il n’est pas fini à 100% car la charge de travail est plutôt élevée par rapport au nombre de périodes que nous disposons (24 périodes) et aussi pour le nombre de personnes par groupe. Il y a plusieurs choses qui pourrait être améliorer, la documentation avec swagger qui n’existe pas pour l’instant, plusieurs routes (celle qui était obligatoire ont été faîtes), un système de gestion de rôles efficace, la mise en place de test unitaire et la dockerisation.

## Personnelle

### Tiago

Ce projet m’a permis d’approfondir mes connaissances acquises lors du module C-295 et m’a fait découvrir la gestion des relations entre les tables. Le projet c’est bien passé et la communication avec mon collègue c’est très bien passé. Je trouve dommage de ne pas avoir eu plus de temps pour aboutir à une api complète.

### Evin

## Planification du projet

Pour la planification nous avons utilisé Trello comme dit dans le point 1 dans analyse, et cela nous a permis de décortiquer le projet pour mieux s’organiser, et vers la fin du projet quand on a compris que ça va être compliqué de tout finir, le fait d’avoir découper le projet auparavant nous a permis de nous réorganiser pour faire les points les plus important pour atteindre notre objectif.

# Webographie

## Docker :

<https://welovedevs.com/fr/articles/docker-node-api/>