|  |
| --- |
| Projet de programmation web Avancée |
|  |

|  |
| --- |
| 15/05/2022 | Promotion : 2024 |

****

**Une image contenant texte, horloge

Description générée automatiquement**

Réalisé par :

Clotaire TCHOUPOU NGAKOU

Hamza SAHMOUNE

Joseph XU

**Gestionnaire d’une bibliothèque d’EFREI**

Table des matières

[INTRODUCTION 2](#_Toc103435176)

[Exigences fonctionnelles de la bibliothèque 3](#_Toc103435177)

[Diagramme de cas d’utilisation 3](#_Toc103435178)

[Exigences non fonctionnelles de la bibliothèque 4](#_Toc103435179)

[L’environnement de travail et Frameworks 4](#_Toc103435180)

[Base de données SQL avec SGBD MySQL 5](#_Toc103435181)

[Cryptage des mots de passe d’authentification avec le module bcrypt 6](#_Toc103435182)

[Gestion des authentifications avec le module JWT (Json Web Tokens) 6](#_Toc103435183)

[Réalisation 7](#_Toc103435184)

[Création de la base de données 7](#_Toc103435185)

[CREATION DU SERVEUR NODE.JS 8](#_Toc103435186)

[CRÉATION DU CLIENT VUE.Js 9](#_Toc103435187)

[FUSION DU SERVEUR ET DU CLIENT EN MODE PRODUCTION AFIN DE LANCER JUSTE LE SERVEUR POUR FAIRE TOURNER L'ENSEMBLE DU PROJET 10](#_Toc103435188)

[CRÉATION DES ROUTES APIs, AUTHENTIFICATION, BIBLIOTHÈQUE, PANIER 10](#_Toc103435189)

[CRÉATION DES INTERFACES ET IMPLEMENTATION DES ROUTE 11](#_Toc103435190)

[Conception du projet 13](#_Toc103435191)

[Problèmes rencontrés 13](#_Toc103435192)

[Solutions apportées 13](#_Toc103435193)

[Voies d’améliorations 13](#_Toc103435194)

[Conclusion 14](#_Toc103435195)

[ANNEXE 15](#_Toc103435196)

[S’inscrire 15](#_Toc103435197)

[S’authentifier 15](#_Toc103435198)

[Rechercher 16](#_Toc103435199)

# INTRODUCTION

Durant cette année nous avons eu l’occasion de participer à des cours en programmation web avancée à la suite de ce que nous avons eu l’année précédente. On a été mené à réaliser un projet dans cette matière dans le but de créer une interface Web avec plusieurs pages afin de s’occuper de la gestion d’une bibliothèque.

Ce site permettra aux étudiants d’EFREI de consulter et emprunter les livres qui sont disponibles au sein de notre bibliothèque. Bien sûr, le responsable de la bibliothèque soit l’administrateur devra avoir la capacité de mettre à jour les différents catalogues et de rajouter de nouveaux livres ou d’en supprimer. Pour ainsi dire, qu’il possède les droits de modification.

Pour ce faire, notre serveur web sera réalisé avec une API (interface de programmation d’application) qui sera codé avec Node.js tout en nous servant du Framework Express.js. Puis, notre base de données sera en MySQL. On utilisera différents langages pour ce projet soit les suivants : HTML, CSS et javascript pour le client ainsi que le serveur. Bien évidemment, pour le côté Client nous avons utilisé Vue.js et le Framework Express.js.

Pour réaliser ce projet, nous avons décidé de disséquer les problèmes et points à réaliser en plusieurs parties. Tout d’abord, dans un premier temps nous avons réalisé un diagramme de cas d’utilisation afin de ne pas dériver et de représenter le comportement qu’aura notre système. Puis nous avons détaillé toutes les contraintes possibles soient les contraintes techniques ou fonctionnelles de ce projet. Pour la base de données, nous l’avons fait en MySQL mais vous trouverez un modèle MCD-MLD réaliser à l’aide de Loopang. Pour conclure, nous finirons par les problèmes rencontrés et les solutions que nous avons apportées.

# Exigences fonctionnelles de la bibliothèque

Pour les exigences fonctionnelles de notre bibliothèque en ligne, nous avons décidé de les retranscrire par un diagramme des cas d’utilisations. Avant tout processus de développement, nous passons du temps sur la réflexion des différents cas, pour mettre au clair le but de notre système c’est-à-dire « servir à quoi ? à qui ».

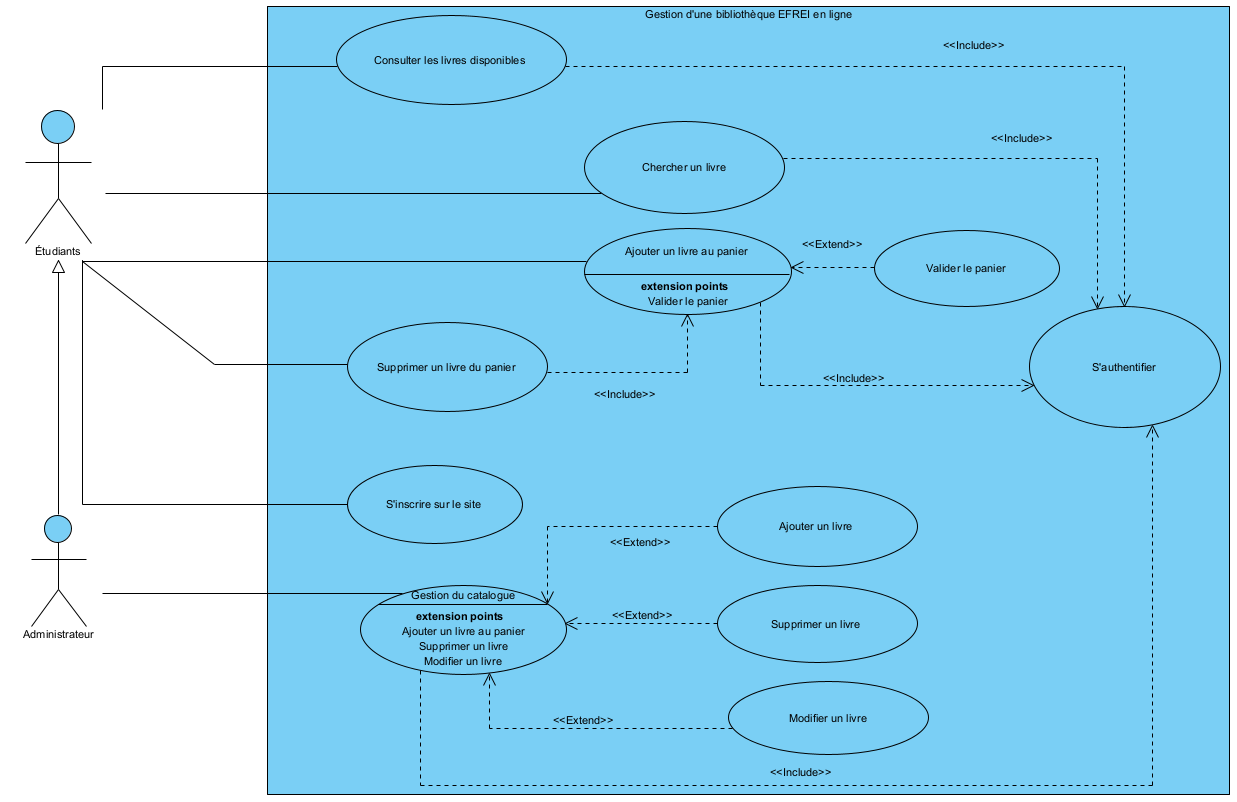
## Diagramme de cas d’utilisation

Dans le diagramme des cas d’utilisations qui se trouve ci-dessous, vous trouverez un cahier des charges sur les fonctionnalités de notre système.

Notre système a pour but de gestionner une bibliothèque en ligne spécialement pour les étudiants de l’EFREI. Alors les étudiants devront tout simplement s’inscrire et rester connectés pour pouvoir emprunter, chercher, consulter les livres qui se trouvent au sein de la bibliothèque soit dans un catalogue. Comme vous pouvez le voir ci-dessous la recherche, la consultation ainsi que la suppression d’un livre du panier correspondent à une extension de l’emprunt d’un livre et tout cela inclus l’authentification.

Pour l’administrateur soit le bibliothécaire, il pourra se connecter afin de gérer la bibliothèque en ayant la possibilité de modifier, supprimer mais également ajouter des livres au sein de cette bibliothèque. Pour se connecter chaque étudiant mais également l’administrateur devront saisir un mot de passe. La recherche de livre se fait en saisissant le titre de l’ouvrage.

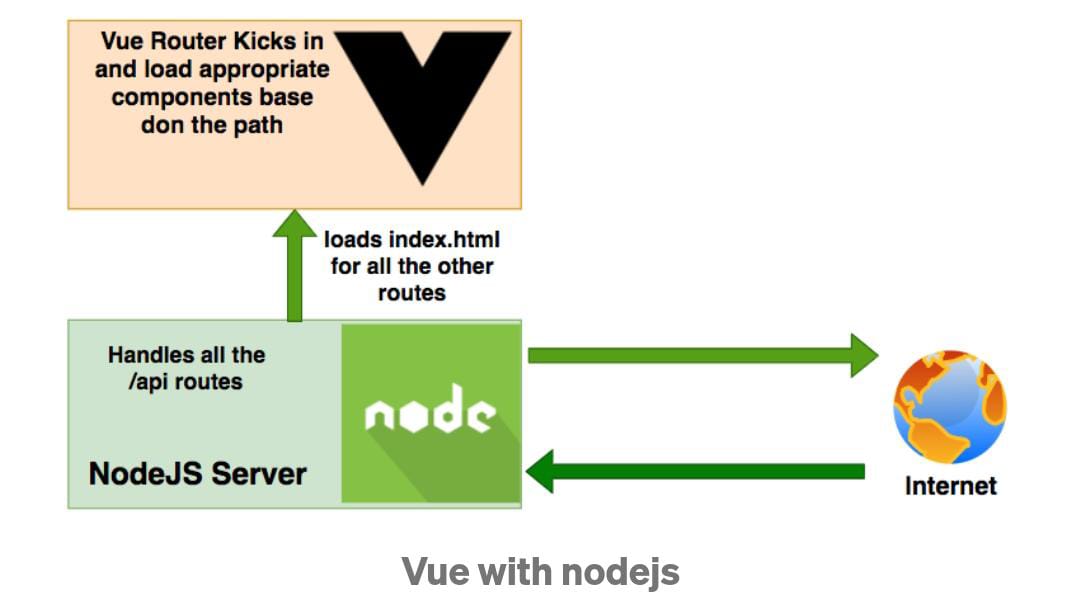
**Figure 1 :** Diagramme des cas d’utilisations pour la gestion d’une bibliothèque à l’aide de Visual Paradigm



# Exigences non fonctionnelles de la bibliothèque

## L’environnement de travail et Frameworks

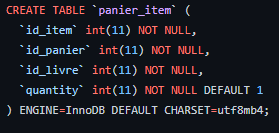
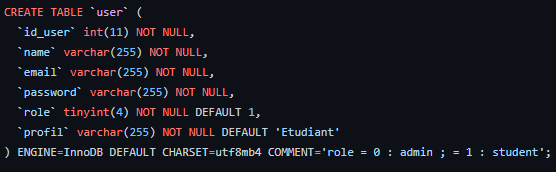
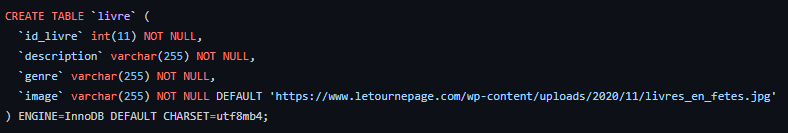
Tout d’abord, nous avons codé le serveur avec le framework backend Node.js tout en utilisant le module Express.js qui a été demandé. La création des interfaces utilisateurs d’applications (API) permet de rendre les interactions utilisateurs assez visibles. Le but étant de faciliter un maximum son utilisation pour ces derniers. Le site est compatible sous différents moteurs de recherches tels que Mozilla Firefox ou Google Chrome. Le client est codé avec le framework frontend Vue.js.



**Figure 2 :** Vue avec nodejs

## Base de données SQL avec SGBD MySQL

Voici les tables principales du projet sous MySQL :

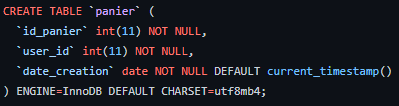


**Figure 3 :** Table panier\_item

**Figure 4 :** Table panier\_item

**Figure 5 :** Table livre

**Figure 6 :** Table user



## Cryptage des mots de passe d’authentification avec le module bcrypt

Le module bcrypt.js est une librairie qui nous permet de faire des haches de mot de passe assez complexe qui différent à chaque fois, ce qui nous permet de stocker les mots de passe de la bonne manière en base de données.

**Code**

## Gestion des authentifications avec le module JWT (Json Web Tokens)

Les authentifications ont été faits avec le module JWT, qui est composé de plusieurs parties et plus précisément trois :

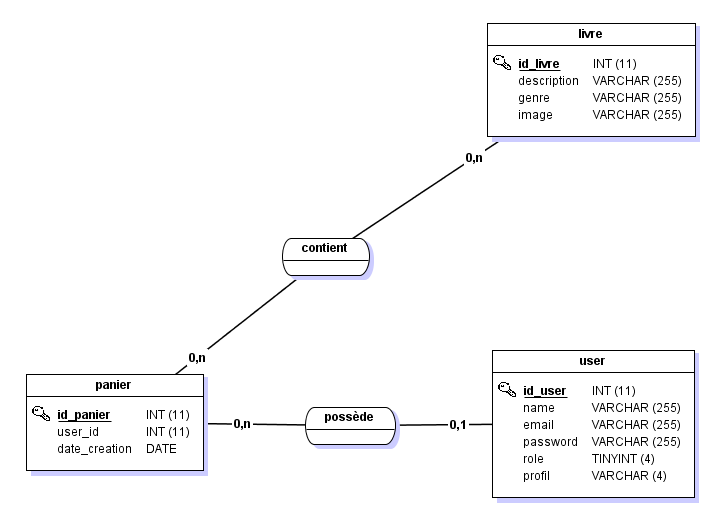
* Un header
* Un payload
* La signature

**Exemple header payload**

# Réalisation

## Création de la base de données

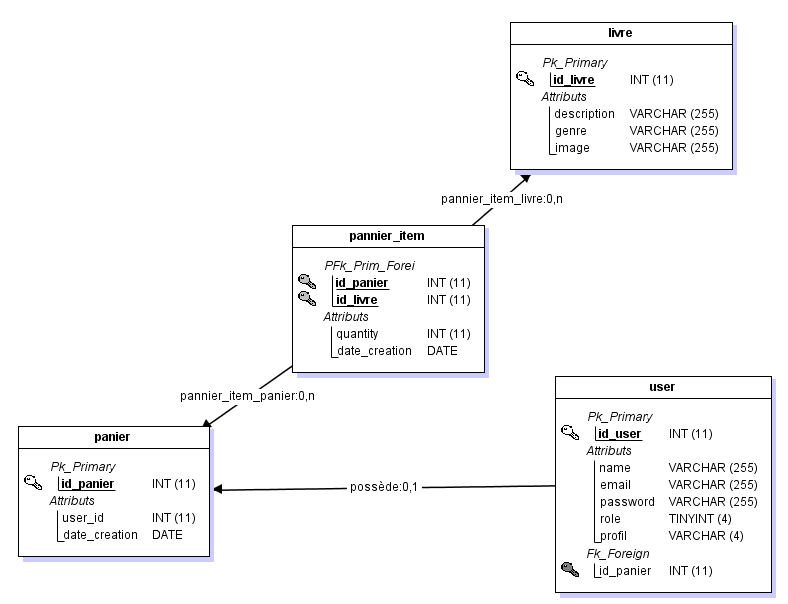
La base de données a été créée sous MySQL, voici les principales tables qui se trouvent au sein de cette dernière, vous trouvez ci-dessous le diagramme MLD et MCD qui ont été réalisé sur JMerise :



**Figure 2 :** MCD des tables sur JMerise

Sur le diagramme, nous remarquons qu’un utilisateur peut posséder 0 à 1 panier s’il souhaite ou non emprunter en ajoutant des livres dans son panier. Par conséquent, le panier peut avoir 0 à plusieurs étudiants, il peut en avoir 0 si aucun étudiant emprunte.

Un panier peut contenir 0 à plusieurs livres en fonction de leur disponibilité, et un livre peut appartenir à 0 ou plusieurs paniers en prenant en compte de la quantité de livre disponible.



**Figure 3 :** MLD des tables sur JMerise

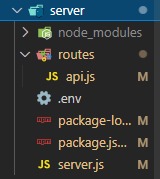
## CREATION DU SERVEUR NODE.JS

La création d’un serveur Node se fait dans notre fichier server.js. On importe le package « http », ce module contient la fonction qui permet de créer le serveur. Notre serveur est lié au port 3000. Comme vous pouvez le voir ci-dessous :



**Figure 4 :** fichier server.js

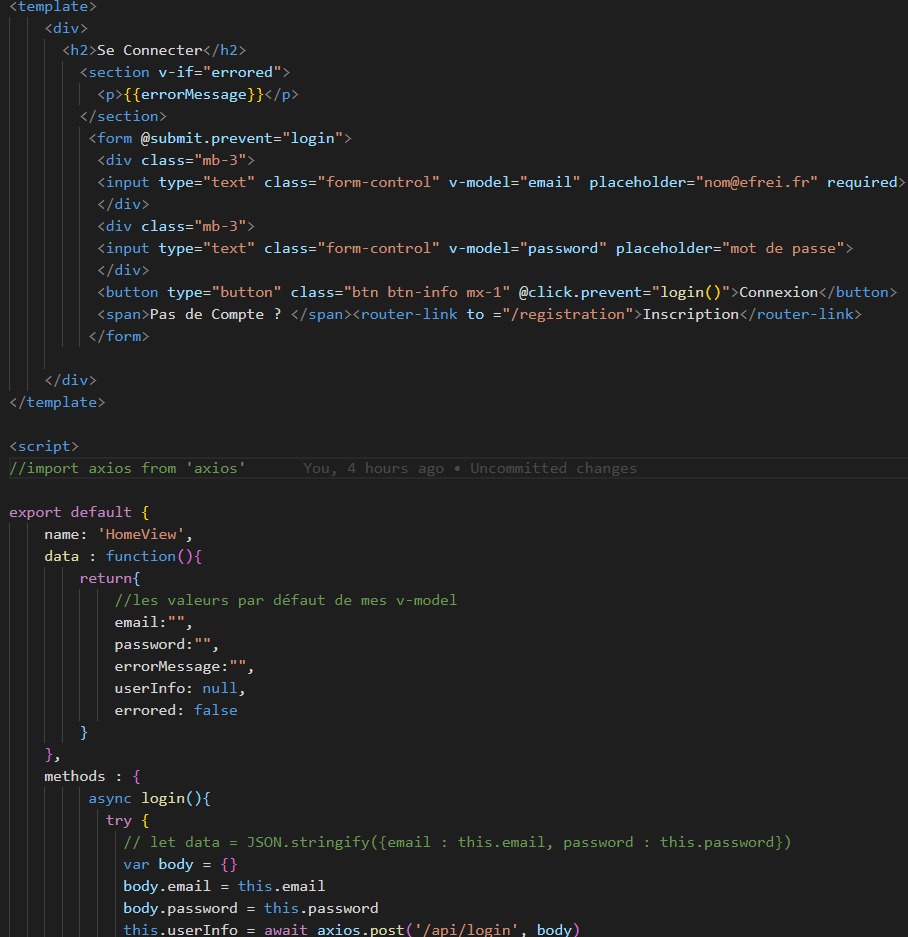
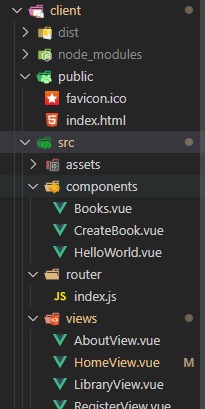
Voici notre serveur :



**Figure 5 :** Serveur

## CRÉATION DU CLIENT VUE.Js

Pour le client, nous l’avons créé à l’aide de vue-cli et vue-router. Pour le cas des différentes requêtes vers le serveur en question nous avons utilisé le module axios. Voici une capture d’écran de notre client et un exemple de vue pour notre page de login :

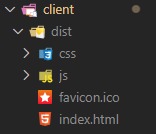
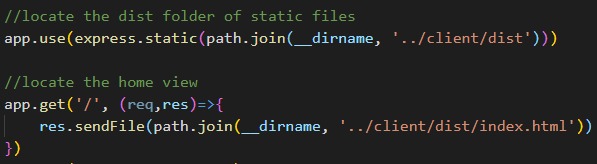


**Figure 6 :** Client

**Figure 7 :** Vue pour la page de login

## FUSION DU SERVEUR ET DU CLIENT EN MODE PRODUCTION AFIN DE LANCER JUSTE LE SERVEUR POUR FAIRE TOURNER L'ENSEMBLE DU PROJET

Tout d’abord il faut exécuter la commande « npm run build » dans le dossier client pour le déployer en mode production. Ceci va générer le dossier « dist » des ressources statiques qui seront accessibles par le serveur node.js via express. Voici quelques captures d’écrans de la fusion du client et du serveur :



**Figure 9 :** Fusion dans le fichier server.js

**Figure 8 :** Dossier « dist » qui sera accessible par le serveur Node

## CRÉATION DES ROUTES APIs, AUTHENTIFICATION, BIBLIOTHÈQUE, PANIER

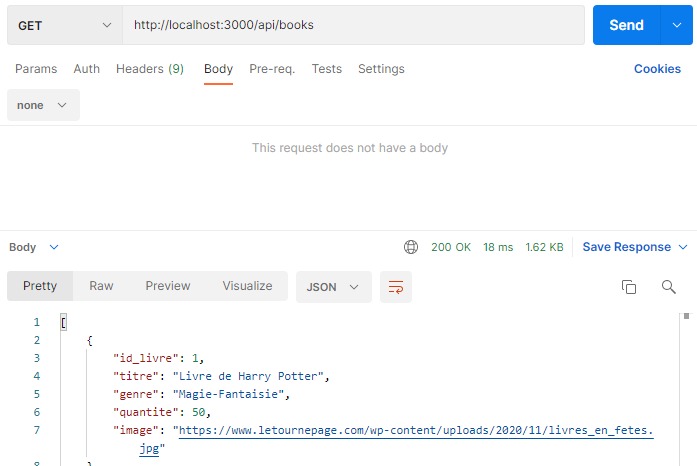
Vous trouverez ci-dessous des captures d’écrans pour la création des routes API soit des routes pour l’authentification, la bibliothèque et le panier :

Une image contenant texte

Description générée automatiquement

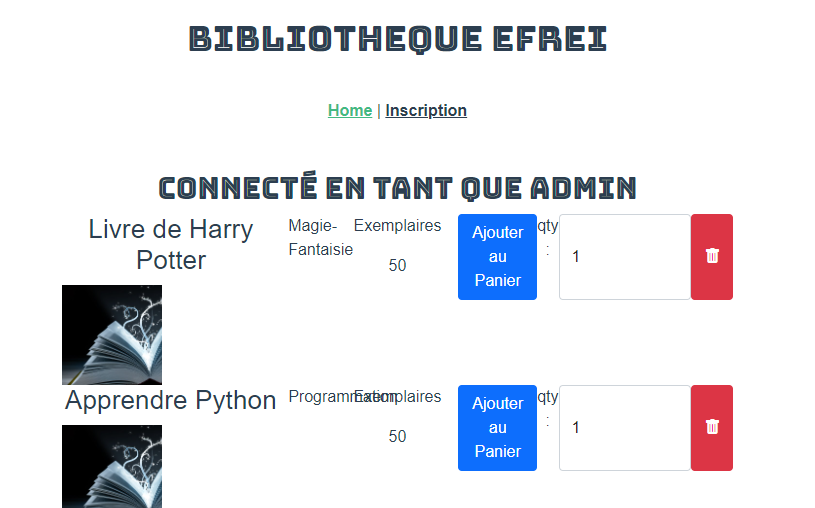
Les routes API

**Figure 10 :** Capture d’écran des routes



**Figure 11 :** route Get Books

## CRÉATION DES INTERFACES ET IMPLEMENTATION DES ROUTE



# Conception du projet

## Problèmes rencontrés

La difficulté majeure était que nous devions utiliser des nouvelles technologies en un temps assez limité

## Solutions apportées

Nous avons fait des recherches sur le Web et principalement avec les documentations officielles.

## Voies d’améliorations

Durant ce projet, nous avons pensé à plusieurs voies d’améliorations possibles pour notre site web de gestion d’une bibliothèque.

* Intégrer un champ indiquant « mot de passe oublié » en cliquant sur ce dernier, ceci emmènera l’étudiant en question vers une page où l’utilisateur aura la capacité de saisir son adresse électronique en l’occurrence son adresse EFREI. Dans le but d’envoyer un mail pour qu’il puisse changer son mot de passe.
* La mise en place d’une page destinée au calendrier des livres à venir qui seront rajoutés dans le catalogue de la bibliothèque.

# Conclusion

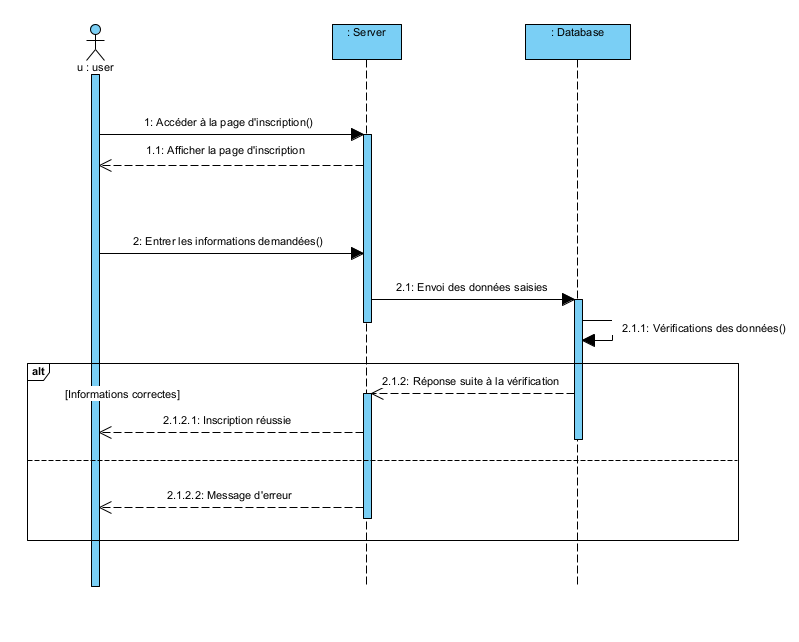
Pour conclure, nous avons eu l’occasion de réaliser un projet en programmation web Avancée, il nous a permis de mettre en œuvre les compétences acquises en cours. Pour ce projet, nous avons également appliqué des connaissances du semestre 5 telles que la création d’un MCD-MLD en base de données et également l’utilisation de MySQL.

Pour ce site web de gestion d’une bibliothèque, on a créé plusieurs pages telles que la page d’inscription ou d’authentification, mais également développé un serveur web avec une interface de programmation d’application (API). Bien évidemment, l’authentification de chaque utilisateur se fait avec des mécanismes de sécurité telles que le hashage des mots de passe. Le cahier des charges a été également respecté, c’est-à-dire que le client a été codé à l’aide de Vue.js, et le serveur avec Node.js en utilisant le module Express.

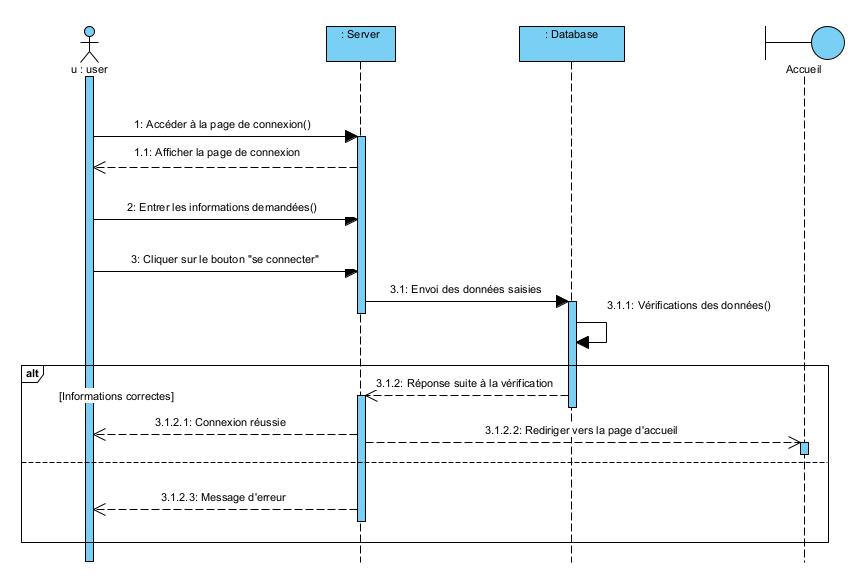
Globalement nous avons pu réaliser toutes les fonctionnalités demandées dans le projet.

# ANNEXE

## S’inscrire



## S’authentifier



## Rechercher

