Samuel Ferragne 2208610  
Liam Brouillard Adjaïlia 2210406  
Médéric Bélec 1950570  
Web et base de données  
420-4A5-MO

Rapport TP2

Travail présenté à  
Sylvain Labranche

Techniques de l'informatique  
Collège Montmorency  
19 avril 2023

Dépôt git : <https://github.com/SamuelMontmorency/4A5_TP2>  
SamuelMontmorency : Samuel Ferragne  
StillCPUEasy : Liam Brouillard Adjaïlia  
Mederic15 : Médéric Bélec

**Compréhension du problème**

1. Requêtes Cours :
   * Pouvoir ajouter des cours, cependant le cours doit beaucoup obligatoirement avoir **un** professeur d’assigné à celui-ci et doit créer un tableau vide d’étudiants. Un cours a les attributs suivants : Titre, Discipline, nb max d’élève, date début, date fin, ID.
   * Pouvoir récupérer les informations liées à un cours grâce à son ID
   * Pouvoir modifier les attributs d’un cours à part son ID.
   * Pouvoir supprimer un cours, s’assurer de retirer le cours de la liste de cours de tous les élèves et tous les profs
2. Requêtes Prof :
   * Pouvoir ajouter un professeur, le tableau de cours est vide à la création. Les profs ont comme attributs : Nom complet, tableau de cours donnés, ID.
   * Pouvoir récupérer les informations liées à un prof grâce à son ID
   * Pouvoir modifier les attributs d’un prof à part son ID.
   * Pouvoir supprimer un professeur, le retirant des cours qu’il enseignait
   * Pouvoir ajouter un cours enseigné à un professeur grâce à l’ID du cours
3. Requête Étudiant :
   * Pouvoir ajouter un étudiant, le tableau de cours est vide à la création. Les étudiants ont comme attributs : Nom complet, tableau de cours suivis, id.
   * Pouvoir récupérer les informations liées à un élève grâce à son ID
   * Pouvoir modifier les attributs d’un élève à part son ID.
   * Pouvoir supprimer un étudiant, le supprimant des cours qu’il avait
   * Pouvoir inscrire un étudiant à un cours existant

**Résultat de l’implémentation**

1. Implémentation des cours

+ Nous avons implémenté l’ajout de nouveaux cours grâce à la route POST. Le nouveau cours a besoin d’un nom et de l’ID du prof. Recevant automatique un tableau vide d’étudiants. Voici un exemple de création de cours : Une image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquement  
- Cependant certains attributs auraient pu être ajouter en plus, que ce soit le nombre max d’étudiant, la date de début, la date de fin, etc.  
+ Nous pouvons récupérer les informations liées à un cours grâce à son ID généré lors de la création en utilisant la route GET. Voici la requête Postman : Une image contenant texte

Description générée automatiquement  
+ Nous avons implémenté la modification des informations relatives aux cours. Ces modifications sont possibles sur Postman avec la route PATCH : Une image contenant texte, capture d’écran, écran, noir

Description générée automatiquementUne image contenant texte

Description générée automatiquement  
- Malheureusement la modification n’est possible qu’avec le nom du cours. Il faudrait donc implémenter de nouvelles fonctions pour permettre de retirer un prof d’un cours donné et retiré un élève d’un cours en plus des fonctions déjà existantes pour les inscrire à différents cours.  
+ L’implémentation finale des cours est la suppression. Grâce à la route DELETE nous pouvons supprimer un cours en le retirant de la liste de cours de tous les profs et élèves puis le supprime de la base de données. Voici la requête Postman : Une image contenant texte

Description générée automatiquement

1. Implémentation des étudiants

+Nous avons implémenté l’ajout d’un étudiant grâce à POST. Le nouvel étudiant a besoin d’un nom, un tableau de cours vide et un ID unique lui est donné ensuite. Exemple : A screenshot of a computer

Description automatically generated with medium confidence

+ Nous pouvons récupérer les informations liées à un étudiant en utilisant la route vers l’ID de l’étudiant et la requête GET. Exemple :

A screenshot of a computer

Description automatically generated with medium confidence

+ Nous avons implémenté la modification du nom des étudiant grâce à PATCH en utilisant la route vers l’ID de l’étudiant et en modifiant l’ancien nom par le nouveau contenu dans le body de la requête. Exemple :

A screenshot of a computer

Description automatically generated with medium confidence

+ Nous avons implémenté la suppression des étudiants grâce à DELETE en utilisant la route vers l’ID de l’étudiant. Il est aussi supprimé des cours qu’il prenait. Exemple :

A screenshot of a computer

Description automatically generated with medium confidence

+Nous avons implémenté l’inscription des étudiants aux cours grâce à la route /inscription puis l’ID de l’étudiant. Le cours auquel l’étudiant est inscrit est celui contenu dans le body de la requête PATCH. Exemple :

A screenshot of a computer

Description automatically generated with medium confidence

1. Implémentation des profs

+Nous avons implémenté l’ajout d’un professeur grâce à POST. Le nouveau professeur a besoin d’un nom, un tableau de cours vide et un ID unique lui est donné ensuite. Exemple :

Une image contenant texte

Description générée automatiquement

**Une image contenant texte

Description générée automatiquement**+ Nous pouvons récupérer les informations liées à un professeur en utilisant la route vers l’ID du professeur et la requête GET. Exemple :

**Une image contenant texte

Description générée automatiquement**+ Nous avons implémenté la modification du nom des professeurs grâce à PATCH en utilisant la route vers l’ID du professeur et en modifiant l’ancien nom par le nouveau contenu dans le body de la requête. Exemple :

+ Nous avons implémenté la suppression des étudiants grâce à DELETE en utilisant la route vers l’ID du professeur. Il est aussi supprimé des cours qu’il prenait. Exemple :

**Une image contenant texte

Description générée automatiquement**

Graphical user interface, text, application

Description automatically generated**Collection MongoDB**

Graphical user interface, text, application

Description automatically generatedText

Description automatically generated

**Journal de bord**

**Samuel Ferragne**

* Mise en place de la db
  + 14 avril (0.5 heure)
* Création du github
  + 14 avril (0.5 heure)
* Initialisation de l’application et création de l’arbre de fichier
  + 17 avril (0.5 heure)
* Création du modèle étudiant
  + 19 avril (0.5 heure)
* Création des routes étudiant
  + 19 avril (0.5 heure)
* Création du controller étudiant
  + 19 avril (2 heures)
* Test des requêtes postman
  + 19 avril (0.5 heure)
* Débogage controller étudiant
  + 19 avril (1 heure)
* Rapport
  + 19 avril (0.5 heure)

**Médéric Bélec**

J’ai commencé à travailler sur la partie professeur durant le cours du 19 avril pour une période d’environ 2 heures. J’ai continué à travailler sur le projet après le cours pour une période d’environ 2h30. Finalement, j’ai complété plus tard dans la soirée pendant une période d’environ 1 heure après avoir pris une pause. J’ai donc passé un total de 5h30.

**Liam Brouillard Adjaïlia**

Mise en place de la db

-19 avril (0.5 heure)Création du model cours

-19 avril (1 heure)  
Création du controller cours

-19 avril (2 heures)  
Création des routes cours

-19 avril (0.5 heure)  
Tests avec Postman

-19 avril (1 heure)  
Corrections de bugs

-19 avril (3 heures)

Création du rapport  
 -19 avril (1 heure)