République Algérienne Démocratique et Populair
Ministère de l'Enseignement Supérieur et de la Recherche Scientifiqu
École Supérieure en Informatique
8 Mai 1945 Sidi Bel Abbès

Rapport du projet

Filière :**I**nformatique

Spécialité : **S**ystèmes d'Information et Web (SIW)

Daresni :une plateforme d'enseignement en ligne dédiée à des leçons de soutient et d'apprentissage de langues

Membres d'équipe :

BEKKHOUCHA Wafaa Fatima Zohra

SIDEL MRABET Malek Aya

NASSANE Marwa

TOUAHRI Sara

YAHIAOUI Meriem

Année Universitaire : 2023 - 2024

Abstract

Avec le développement rapide de la technologie numérique, l'enseignement en ligne est devenu une composante essentielle de l'éducation moderne. Cependant, il existe une lacune dans les plateformes spécifiquement conçues pour les cours particuliers à l'échelle nationale en Algérie. Le projet "DARESNI" a été conçu pour combler ce manque en offrant une plateforme dédiée aux leçons de soutien pour les élèves du CEM et du Lycée.

La plateforme "DARESNI" permet aux enseignants et aux élèves d'interagir de manière efficace dans un environnement en ligne convivial et intuitif. Elle se distingue par une fonctionnalité d'apprentissage des langues, actuellement centrée sur l'apprentissage de la langue arabe, offrant ainsi aux utilisateurs la possibilité d'améliorer leurs compétences linguistiques en plus des autres services éducatifs.

Le projet utilise une approche "Cloud-Native" avec une architecture de microservices. Cette architecture modulaire inclut plusieurs services tels que la gestion des comptes, l'authentification et les droits d'accès, la gestion des cours, et bien d'autres. Chaque microservice est conçu pour être indépendant, facilitant ainsi la maintenance et l'évolutivité de la plateforme.

Ce rapport présente en détail le projet "DARESNI", ses objectifs, sa conception, son architecture et ses fonctionnalités, illustrant comment cette plateforme innovante répond aux besoins spécifiques de l'enseignement en ligne en Algérie.

Table des matières

Table des figures

Chapitre 1

Introduction

Avec le développement rapide de la technologie numérique, l'enseignement en ligne est devenu une partie intégrante de l'éducation. Plusieurs plateformes d'enseignement en ligne ont été créées et sont largement utilisées dans le monde entier. Cependant, il existe toujours un manque de plateformes spécifiquement conçues pour les cours particuliers à l'échelle nationale en Algérie.

Dans ce contexte, le projet "DARESNI" entre en jeu. Il s'agit d'une plateforme d'enseignement en ligne dédiée à ces leçons de soutien. L'objectif de cette plateforme est de combler ce manque en offrant un espace en ligne où les enseignants et les élèves (CEM et Lycée) peuvent interagir efficacement dans un environnement en ligne convivial et intuitif.

Un des atouts majeurs de notre plateforme est qu'elle offre une fonctionnalité d'apprentissage des langues, actuellement centrée sur la langue arabe. Cela permet aux utilisateurs d'améliorer leurs compétences linguistiques tout en bénéficiant d'autres services éducatifs.

La plateforme doit être réalisée en utilisant l'approche "Cloud-Native" ou ce que nous entendons par l'architecture micro-service. La plateforme s'articule sur plusieurs modules ou micro-services, dont la gestion des comptes, l'authentification et les droits d'accès, la gestion des cours, les groupes d'étudiants, les profs, les spécialités, etc.

Ce rapport présentera en détail le projet "DARESNI", ses objectifs, sa conception, son architecture et ses fonctionnalités.

FIGURE 1.1 – Daresni Logo

Chapitre 2

Fonctionnement de la plateforme

Étape 1 : Inscription de l'Enseignant

— L'Enseignant sur la plateforme en insérant leur nom,prénom, adresse e-mail, mot de passe et nom et leur numéro de téléphone et son CV .

Étape 2 : Validation de l'enseignant par l'Administrateur

- L'administrateur recevra une notification indiquant qu'un nouvel utilisateur s'est inscrit.

— Le cv de l'enseignant est examiné attentivement par l'administrateur de la plateforme.

— L'administrateur évalue les compétences, l'expérience et les qualifications de l'enseignant pour vérifier s'il répond aux critères établis par la plateforme.

— Si l'enseignant est validé par l'administrateur il pourra commencer à utiliser la plateforme.

Étape 3 : Création des sessions par l'enseignant

— L'enseignant va créer des sessions en précisant le nom de la matière, l'année, la spécialité, le nombre maximum d'étudiants qui peuvent s'inscrire, le prix, le nombre de sessions par semaine, la durée du programme (en nombre de semaines) et la durée de chaque session (le nombres d'heures par séance).

Étape 4 : Validation des sessions par l'administrateur

- L'adminstrateur recevra une notification indiquant q'un enseignant veut créer une session

— L'administrateur doit vérifier les séssions créés et les valider.

Étape 5 : Création des groupes

— Aprés l'approbation de l'administrateur ,l'enseignant pourra créer les groupes,où il va préciser le nom du groupe,la date début et la deadline.

Étape 6 : Navigation et inscription d'un étudiant dans un groupe

— L'étudiant peut filtrer selon l'année, sa spécialité, son niveau et la matière pour laquelle il souhaite suivre des cours de soutien et le prix aussi.

— Ensuite, les sessions qui correspondent à sa recherche vont apparaître, contenant les groupes disponibles, et il pourra en choisir un pour s'inscrire en indiquant son nom,prénom, adresse e-mail, mot de passe et leur numéro de téléphone .

— L'étudiant doit payer pour valider son inscription dans le groupe.

Étape 7 :Utilisation les fonctionnalités offertes par la plateforme.

— L'enseignant peux consulter son emplois du temps (même dans son google calender associé à son email),ses groupes etc...

- L'enseignant peux faire des cours en visioconférence (à travers google meet).

- L'enseignant peut uploader ses cours ainsi que tous les fichiers liés à sa formation (séries d'exercices, etc...) et les vidéos des cours effectués en visioconférence pour qu'ils puissent être visualisés par les étudiants.

- Utilisation de l'espace messagerie et forum.

Notre plateforme propose aussi une fonctionnalité d'apprentissage de langues. Voyons ci-dessous comment on peut l'utiliser :

Étape 1 : Choisir la langue que l'étudiant souhaite apprendre.

— Une fois que l'étudiant accède à cette fonctionnalité, il peut choisir la langue qu'il veut (arabe,anglais ou français) et il peut aussi choisir entre faire son apprentissage dans la grammaire ou le vocabulaire.

Étape 2 :Le déroulement du processus d'apprentissage.

- Il y a plusieurs niveaux, et chaque niveau contient plusieurs étapes qui renferment des informations adaptées à ce niveau.

Étape 3 : Passer d'un niveau à un autre.

- Pour que l'étudiant passe d'un niveau à un autre, il doit passer un examen qui sera validé par l'administrateur.

Chapitre 3

Conception du Système

3.1 Introduction

Dans cette partie, nous allons vous présenter les divers microservices qui opèrent conjointement pour créer notre application de gestion des externalisations. Chacun des microservices est spécialement conçu pour fonctionner de manière indépendante afin de fournir la plus grande flexibilité, maintenabilité et extensibilité possible pour la solution intégrée. Nous allons parler des microservices les plus importants qui contribuent au bon fonctionnement de l'application avec une brève introduction à leurs entités et objectifs.

3.1.1 Microservice d'Authentification

Ce microservice gère les fonctionnalités d'authentification et d'autorisation sur notre plateforme. Il comprend les entités suivantes :

CreateAdmin : L'entité "createAdmin" est utilisée pour la création d'un administrateur. Elle sert de structure de validation et de transfert des informations nécessaires à cette création. Cette entité garantit que les données de l'administrateur, telles que l'email, le mot de passe et le rôle, respectent des contraintes spécifiques de format et de complétude. En utilisant des décorateurs de validation, l'entité "CreateAdmin" assure que chaque champ est correctement formaté avant d'être traité par le système. Les relations entre les propriétés de l'entité permettent une gestion rigoureuse des données de l'administrateur, facilitant ainsi l'intégrité et la sécurité des opérations d'administration.

Login : L'entité "Login" est utilisée pour gérer les opérations de connexion et de validation de l'authentification d'un utilisateur. Cette entité garantit que les données de connexion, telles que l'email et le mot de passe, respectent des contraintes spécifiques de format et de complétude. En utilisant des décorateurs de validation, l'entité "Login" assure que chaque champ est correctement formaté avant d'être traité par le système. Les relations entre les propriétés de l'entité

permettent une gestion rigoureuse des données de connexion, facilitant ainsi l'intégrité et la sécurité des opérations d'authentification.

Prof : L'entité "Prof" pour gérer les informations relatives aux professeurs. Elle s'assure de la présence des informations nécessaires telles que le nom, le prénom, l'email, le numéro de téléphone, le CV et la photo. Cette entité garantit que les informations du professeur respectent des contraintes spécifiques de format et de complétude, tout en permettant certaines options facultatives comme le numéro de téléphone, le CV et la photo. En utilisant des décorateurs de validation, l'entité "Prof" assure que chaque champ est correctement formaté avant d'être enregistré dans le système, facilitant ainsi une gestion cohérente et sécurisée des données des professeurs.

Student : L'entité "Student" utilisé pour la gestion des informations relatives aux étudiants. Elle sert à structurer la validation et le transfert des données telles que le nom, le prénom, l'email, le niveau, l'année d'étude, la spécialité et le niveau de langue. Cette entité garantit que les informations de l'étudiant respectent des contraintes spécifiques de format et de complétude, tout en permettant certaines options facultatives comme le numéro de téléphone. En utilisant des décorateurs de validation, l'entité "Student" assure que chaque champ est correctement formaté avant d'être enregistré dans le système, facilitant ainsi une gestion cohérente et sécurisée des données des étudiants.

UpdateProf : L'entité "UpdateProf" sert de structure flexible permettant de valider et de transférer uniquement les données à modifier, telles que le nom, le prénom, le numéro de téléphone, le CV, la photo et une description. Cette entité utilise des décorateurs de validation pour assurer que chaque champ optionnel est correctement formaté avant d'être mis à jour dans le système. Les relations entre les propriétés de l'entité permettent une gestion précise et sécurisée des modifications apportées aux informations des professeurs.

UpdateStudent : L'entité "UpdateStudent" est utilisée pour la mise à jour des informations des étudiants. Elle sert de structurer uniquement les transfers de données à modifier, telles que le nom, le prénom, le numéro de téléphone, le niveau, l'année d'étude et la spécialité. Cette entité utilise des décorateurs de validation pour assurer que chaque champ optionnel est correctement formaté avant d'être mis à jour dans le système. Les relations entre les propriétés de l'entité permettent une gestion précise et sécurisée des modifications apportées aux informations des étudiants.

UpdateUser : L'entité "UpdateUser" est utilisée pour la mise à jour des informations des utilisateurs. Elle sert de structure flexible permettant de valider et de transférer uniquement les données à modifier, telles que l'email et le mot de passe. Cette entité utilise des décorateurs de validation pour assurer que chaque champ optionnel est correctement formaté avant d'être mis à jour dans le système.

Les relations entre les propriétés de l'entité permettent une gestion précise et sécurisée des modifications apportées aux informations des utilisateurs.

User : L'entité "User" gère les informations de création d'un utilisateur telles que l'email, le mot de passe, le rôle et l'état d'activation. Cette entité garantit que les informations de l'utilisateur respectent des contraintes spécifiques de format et de complétude. En utilisant des décorateurs de validation, l'entité "User" assure que chaque champ est correctement formaté avant d'être enregistré dans le système. Les relations entre les propriétés de l'entité permettent une gestion rigoureuse des données de l'utilisateur, facilitant ainsi l'intégrité et la sécurité des opérations d'administration.

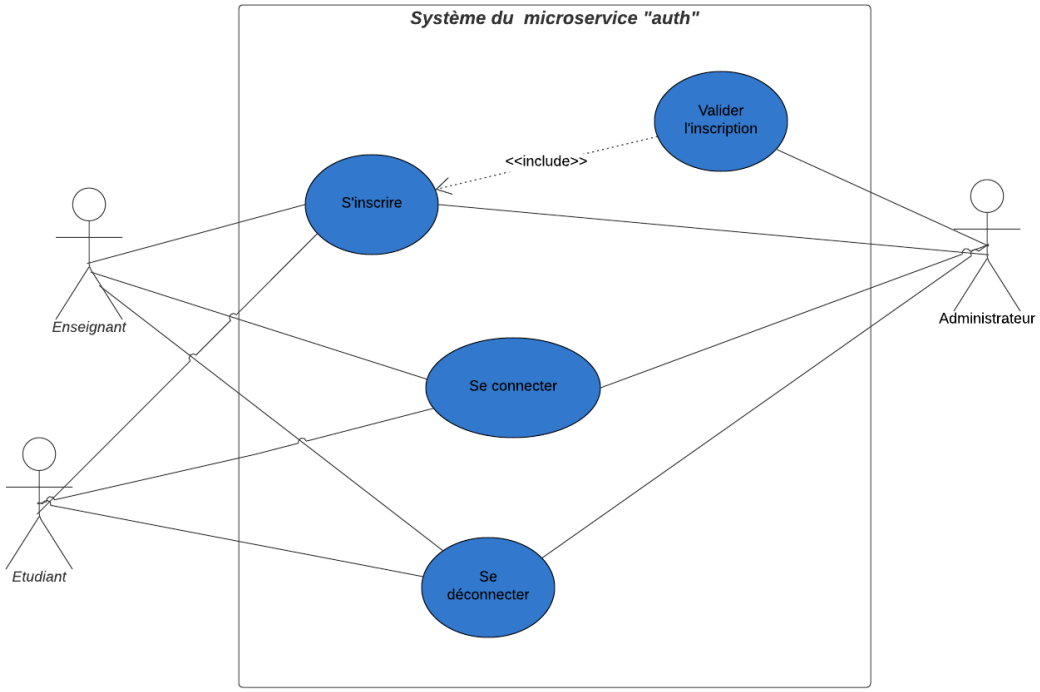

FIGURE 3.1 – Diagramme de cas d'utilisation du micro-service d'Authentification

3.1.2 Microservice gestion de groupes "MS-GROUP"

Ce microservice est responsable de la gestion de la création des groupes de la part des enseignants,et les entités de ce service se présntent comme suit :

GroupContainer : L'entité "GroupeContainer" représente la session,elle sert d'unité organisationnelle centrale pour la gestion des groupes éducatifs. Elle offre

un moyen structuré de définir des modules d'apprentissage, de spécifier le nombre maximale des étudiants et de fixer les prix. Les relations entre les entités GroupContainer et group permettent une gestion granulaire des inscriptions individuelles des étudiants et du suivi de leurs progrès au sein de chaque groupe.

Groupe : L'entité "Groupe" Représente un groupe spécifique d'étudiants inscrits à un module d'apprentissage particulier.Elle sert de conteneur pour gérer des informations détaillées sur le groupe et ses activités.

Elle offre un moyen complet de gérer des groupes individuels. Elle capture des détails tels que les horaires des sessions, les données d'inscription et les ressources ou enregistrements pertinents. En associant Group à GroupContainer, on va créer une structure hiérarchique pour organiser qui va faciliter le suivi des étudiants, et le suivi des progrès dans différents groupes et modules.

FIGURE 3.2 – Diagramme de cas d'utilisation du microservice "MS-GROUP"

3.1.3 Microservice du paiment "MS-PAIMENT"

Ce microservice gère le paiment des étudiants,il comprend l'entité suivante :

StudentAccount : L'entité "StudentAccount", est l'élément principal de la prise en charge des transactions financières de la plateforme,L'entité StudentAccount est une représentation dans notre système de gestion des paiements qui encapsule les informations relatives aux comptes étudiants.

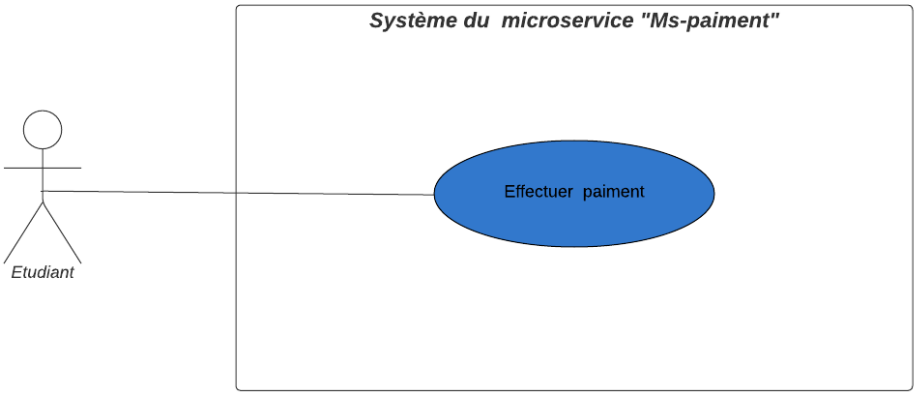

FIGURE 3.3 – Diagramme de cas d'utilisation du microservice "MS-PAIMENT"

3.1.4 Microservice de la gestion des notifications "MS-Notification"

Ce microservice, spécialement conçu pour orchestrer la diffusion efficace des notifications au sein de notre application,il se compose de plusieurs entités clés dédiées à la gestion des flux de données en temps réel. En son sein, nous trouvons le module **"KafkaModule"**, qui abrite le service de consommation "ConsumerService". Ce dernier est chargé de recevoir et de traiter les messages Kafka, assurant ainsi une communication fluide et asynchrone entre les différents composants de notre système. Grâce à l'utilisation de Mongoose, nous structurons et validons les données relatives aux notifications, telles que les notifications de création de sessions et les détails des professeurs, garantissant ainsi leur intégrité et leur cohérence.

3.1.5 Microservice de Forum et Messagerie "Ms-ForumMessagerie"

Ce microservice il est divisé en deux sous-microservices : **MS-Forum et MS-Messagerie**

Le microservice Forum :

Ce sous-microservice gère les publications et les commentaires il contient les entités suivantes :

Comments : L'entité 'Comment' permet aux utilisateurs d'interagir de manière significative avec les publications. En enregistrant le contenu des commentaires, les informations sur leurs auteurs, ainsi que la date et l'heure de leur création, elle offre un moyen essentiel pour exprimer des opinions, engager des discussions et favoriser l'interaction au sein de notre plateforme. En reliant chaque commentaire à une publication spécifique, elle crée un environnement participatif où les utilisateurs peuvent réagir et donner leur avis, enrichissant ainsi l'expérience utilisateur. Cette fonctionnalité est essentielle pour stimuler l'engagement des utilisateurs et favoriser la création d'une communauté active et participative autour de notre plateforme, tout en offrant aux administrateurs une visibilité précieuse sur les réactions générées par le contenu publié.

Forums : L'entité 'Forum' nous aide à offrir un espace organisé où les utilisateurs peuvent partager leurs idées, poser des questions et interagir les uns avec les autres. Pour ce faire, elle utilise un identifiant de session pour distinguer les différentes instances de forums, garantissant une gestion sécurisée des discussions, tandis que le suivi de la date de création de chaque forum permet de conserver un historique temporel précis des échanges. De plus, la capacité à lier les discussions aux publications pertinentes facilite la navigation et la compréhension du contexte des discussions, favorisant ainsi une expérience utilisateur enrichissante et collaborative.

Post : L'entité 'Post' enregistre les publications des utilisateurs, elle offre un espace essentiel où ces derniers peuvent partager du contenu, poser des questions et engager des discussions. Chaque publication est caractérisée par un titre, un contenu, les informations sur son auteur, une référence au forum auquel elle est associée, ainsi qu'une date de création. En reliant chaque publication à un forum spécifique, elle permet une organisation claire des discussions et favorise l'interaction entre les utilisateurs. De plus, en fournissant une liste de références aux commentaires associés à chaque publication, elle facilite le suivi des discussions et la participation des utilisateurs.

Users : L'entité User représente un élément essentiel de notre système, offrant la fonctionnalité de gestion des utilisateurs. Enregistrant les informations d'identification telles que le nom d'utilisateur, l'adresse e-mail et le mot de passe, elle assure l'authentification et la sécurité des utilisateurs lors de leur accès à la

plateforme. Chaque utilisateur est caractérisé par un nom d'utilisateur et une adresse e-mail uniques, garantissant ainsi l'unicité de leur identité au sein du système. De plus, en enregistrant la date de création de chaque utilisateur, elle offre une traçabilité des comptes et facilite la gestion des activités des utilisateurs sur la plateforme. En résumé, l'entité User joue un rôle central dans la sécurisation et la gestion des utilisateurs, assurant ainsi une expérience utilisateur fiable et sécurisée sur notre plateforme.

Le microservice Messagerie : Grâce à ce microservice, les utilisateurs peuvent échanger des messages de manière efficace et fiable, favorisant ainsi la collaboration et la coordination entre eux.Les entités présentes sont :

ChatGroup : L'entité 'ChatGroup' fournit une structure organisée pour la gestion des groupes de discussion. Enregistrant des informations telles que le nom du groupe de discussion et l'identifiant de session associé, il facilite la création et la gestion de plusieurs discussions simultanées au sein de notre plateforme. De plus, en maintenant une liste de références aux messages échangés dans chaque groupe, il permet un suivi précis des interactions entre les utilisateurs. En centralisant la gestion des discussions de groupe.

Message : L'entité 'Message' permet aux utilisateurs d'échanger des messages au sein des groupes de discussion. Enregistrant le contenu de chaque message, ainsi que le nom de l'utilisateur qui l'a envoyé et l'identifiant du groupe de discussion auquel il est associé, . En capturant les interactions en temps réel entre les utilisateurs à l'aide de Socket.IO , le modèle Message facilite la communication et la collaboration au sein de notre plateforme.

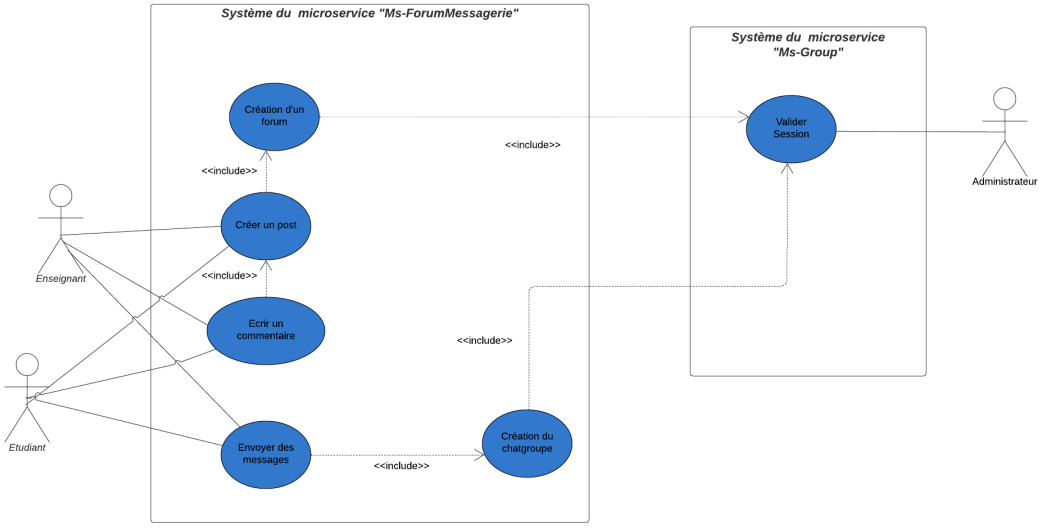

FIGURE 3.4 – Diagramme de cas d'utilisation du micro-service" Ms-ForumMessagerie "

3.1.6 Microservice filtrage 'Ms-Filtrage'

Ce microservice est conçu pour gérer les opérations de recherches des sessions/groupes par filtrage effectuer par les étudiants

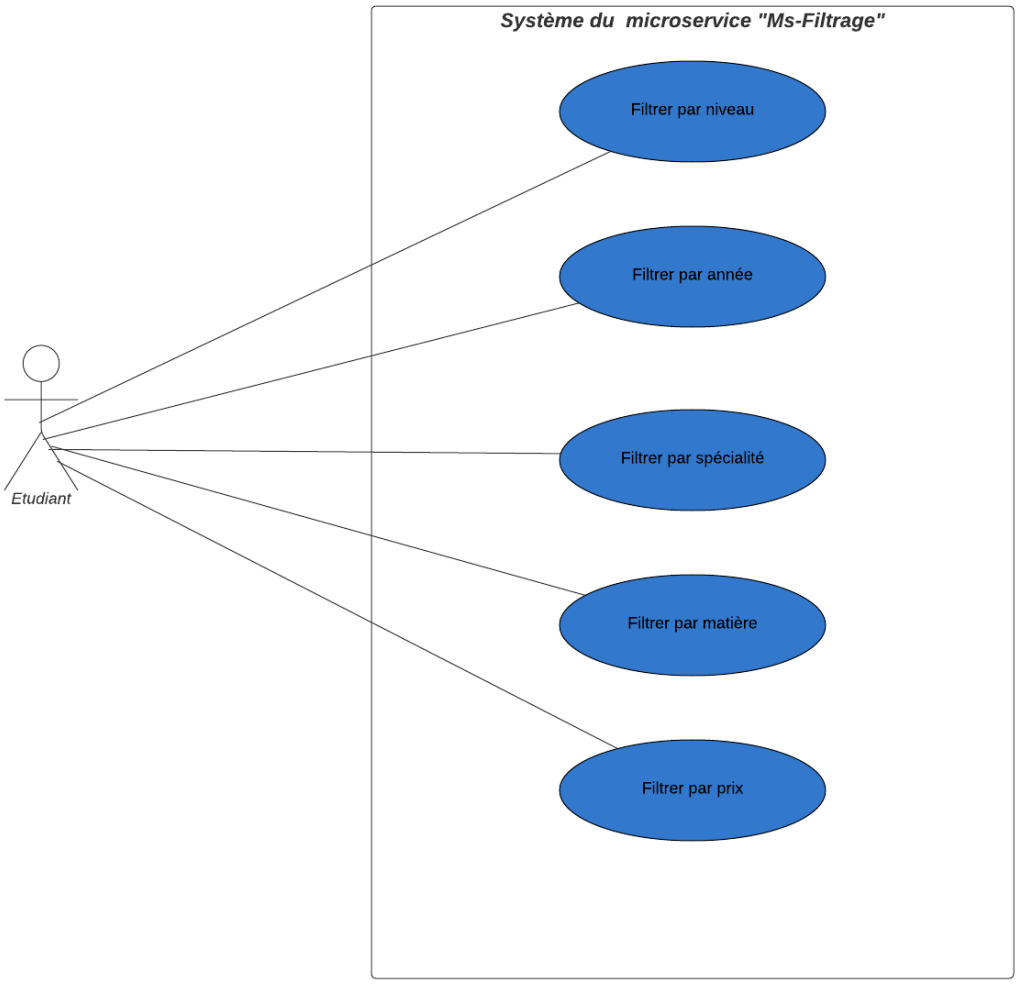

FIGURE 3.5 – Diagramme de cas d'utilisation du micro-service" Ms-Filtrage"

3.1.7 Microservice Apprentissage de langues 'Ms-ApprentissageLangues'

Le microservice d'apprentissage de langues, nommé 'Ms-ApprentissageLangues', est conçu pour offrir des fonctionnalités avancées d'apprentissage de langues (arabe,français et anglais) aux utilisateurs, on utilise les entités suivantes :

Language : Cette entité contient : - L'énumération Linguistic définit les différents aspects linguistiques couverts par un niveau d'apprentissage, tels que le vocabulaire, la grammaire ou une combinaison des deux. Elle permet de spécifier le type de contenu linguistique enseigné à chaque niveau. - L'énumération Language définit les différentes langues prises en charge par la plateforme. Elle permet de spécifier la langue d'enseignement pour les différents niveaux et étapes d'apprentissage.

Level : L'entité Level permet de structurer et de gérer les différents niveaux d'apprentissage au sein de notre plateforme d'apprentissage des langues. Elle enregistre des informations cruciales telles que le nom du niveau, les détails linguistiques associés, les étapes d'apprentissage à compléter, le chemin vers l'examen final, et les identifiants des étudiants inscrits. En centralisant ces données, l'entité Level facilite l'organisation pédagogique et le suivi de la progression des étudiants.

StudentInfo : L'entité StudentInfo est utilisé pour transférer les détails d'un niveau d'apprentissage spécifique pour un étudiant. Elle contient des informations sur le niveau, telles que l'identifiant, le nom, les détails linguistiques, les étapes, le chemin de l'examen et l'étape actuelle de l'étudiant. Cette entité facilite la transmission des données entre les différentes couches de l'application.

Step : L'entité Step représente les différentes étapes ou modules d'apprentissage au sein d'un niveau. Elle enregistre des informations détaillées sur chaque étape, telles que le titre, la description, les règles, des exemples, des questions et des réponses. Cette entité permet de structurer les contenus éducatifs et de suivre la progression des étudiants à travers les différentes étapes.

Student : L'entité Student représente les étudiants inscrits sur la plateforme d'apprentissage des langues. Elle enregistre les informations clés sur chaque étudiant, telles que leur identifiant, le chemin vers leurs solutions d'examen, leur adresse e-mail, leur étape actuelle dans le parcours d'apprentissage et l'identifiant du niveau actuel. Cette entité permet de suivre la progression de chaque étudiant et de gérer leurs interactions avec les contenus éducatifs.

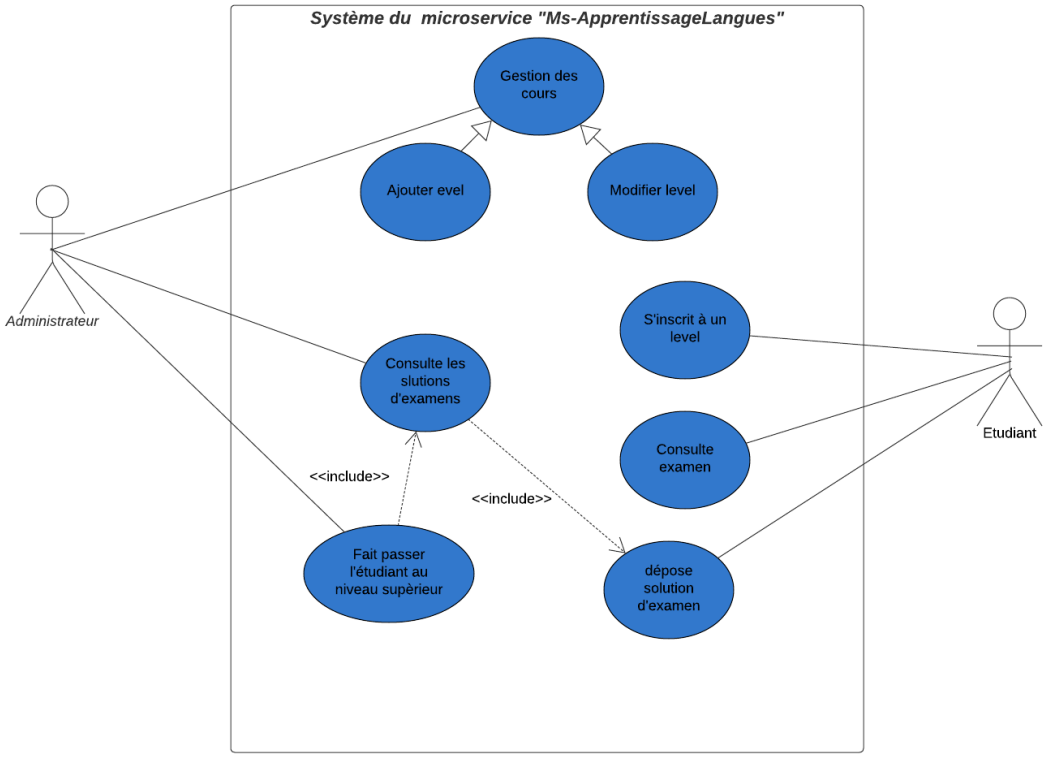

FIGURE 3.6 – Diagramme de cas d'utilisation du microservice "Ms-ApprentissageLangues"

3.2 Communications entre les microservices

3.2.1 Microservice' Authentification' :

Ms-Notification : - A chaque nouvelle inscription d'un enseignant une notification sera envoyée à l'admin.

3.2.2 Microservice 'Ms-Groupe' :

- Le microservice 'Ms-Groupe'est responsable de la gestion des groupes et sessions de notre plateforme,il communique avec :

ms-auth :

- Un enseignant doit être anthentifié pour créer les sessions/groupes. - Pour que l'étudiant accède il doit s'authentifier.

Ms-paiment :

- Pour valider l'inscription de l'étudiant à un groupe il faut s'assurer qu'il a payé.

Ms-ForumMessagerie :

- Quand l'administrateur valide une session, un chat de groupe est automatiquement créé et associé à cette session.

Ms-Notification :

- Quand un enseignant crée une session une notification sera envoyée a l'admin .

- Aprés la validation de la session l'enseignant va avoir une notification.

- Quand l'étudiant va s'inscrire dans un groupe l'enseignant va recevoir une notification.

- Quand l'enseignant dépose un document les étudiant vont recevoir une notification.

3.2.3 Microservice 'Ms-Filtrage' :

Ms-Group :

- Aprés que l'étudiant effectue son filtrage comme il le souhaite les groupes disponible dans sa recherche vont apparaitre.

3.3 Protocoles de communication

Nos microservices utilisent divers protocoles de communication en fonction des besoins spécifiques. Les deux principaux types de communication sont :

Communication Synchrone : HTTP/REST : Le protocole HTTP avec des API RESTful est largement utilisé pour les communications synchrones. Cela signifie qu'un microservice envoie une requête à un autre et attend une réponse avant de continuer son traitement. Les API REST sont bien adaptées pour les appels directs et immédiats où une réponse rapide est nécessaire.

Communication Asynchrone Pour les communications asynchrones, On a utilisé Apache Kafka sont . Kafka permet aux microservices de publier et de s'abonner à des flux de données en temps réel, facilitant une communication découplée où les services ne dépendent pas d'une réponse immédiate.

Chapitre 4

Conception de l'Interface Utilisateur

4.1 Introduction :

Dans le développement de logiciels et d'applications, il est essentiel de prêter une attention particulière à la conception de l'interface utilisateur (UI). Elle se rapporte à la conception de interfaces par lesquelles les utilisateurs communiquent avec un système. L'objectif d'une interface utilisateur de qualité est de procurer une expérience utilisateur (UX) qui soit intuitive, performante et plaisante. La conception doit servir à la fois l'esthétique et la fonctionnalité, permettant ainsi une interaction fluide entre l'utilisateur et l'application.

Le but principal de concevoir l'interface utilisateur est d'assurer une interaction aussi simple et efficace que possible. Cela nécessite une connaissance approfondie des besoins des utilisateurs, ainsi que la maîtrise des principes du design ergonomique et de l'accessibilité. Une interface utilisateur bien pensée peut grandement améliorer la satisfaction des utilisateurs, accroître leur engagement et minimiser les erreurs.

4.2 Exigences des Utilisateurs

Pour concevoir l'interface utilisateur de la plateforme d'enseignement en ligne Daresni, dédiée aux leçons de soutien et à l'apprentissage des langues, il est crucial de comprendre les attentes et besoins des utilisateurs. Après avoir effectué des recherches détaillées, nous avons pu cerner les besoins essentiels des utilisateurs, notamment en termes de convivialité, de recherche efficace de cours et d'enseignants qualifiés, de communication fluide entre les utilisateurs et également la sécurité des transactions.

Afin de mieux cerner les utilisateurs et orienter la conception de l'interface utilisateur, nous avons élaboré des profils personnifiés se basant sur leurs caractéristiques démographiques, compétences pédagogiques et motivations. Les personnes concernées comprennent des enseignants qui fournissent un soutien

scolaire et dispensent des cours de langues, ainsi que des étudiants cherchant à améliorer leurs compétences linguistiques ou à obtenir une aide académique. En outre, il y a aussi les administrateurs de la plateforme responsables de l'organisation et Ces individus nous aident à adapter la conception afin de répondre aux besoins spécifiques de chaque groupe d'utilisateurs, assurant ainsi une expérience utilisateur optimale et personnalisée pour chaque profil.

4.3 Wireframes et Mockups de l'Interface Utilisateur

Daresni, une plateforme d'enseignement en ligne dédiée aux leçons de soutien et à l'apprentissage des langues, utilise les wireframes et les mockups comme outils visuels essentiels pour représenter la structure, la disposition et les fonctionnalités de son interface utilisateur. Nous avons créé des wireframes détaillés pour chaque page, prenant en compte les besoins et les exigences des utilisateurs.

Ces wireframes illustrent la disposition précise de tous les éléments tels que le menu de navigation, les formulaires et les options de filtrage. Les wireframes ont été utilisés comme point de départ pour développer des mockups interactifs qui reproduisent précisément l'apparence visuelle de l'interface, y compris les couleurs, typographies, images et icônes.

Les principales pages, telles que la page d'inscription des enseignants, de création des sessions et de navigation et d'inscription des étudiants ont été élaborées dans le but de fournir une expérience utilisateur intuitive et plaisante. Dans ce processus, l'administrateur s'occupe de l'enregistrement et de la vérification des enseignants, ainsi que de la création et de la validation des sessions. Il est également en charge de créer les groupes et d'inscrire les étudiants. En plus de cela, le système propose diverses fonctionnalités telles que la visioconf Les étudiants ont la possibilité de choisir leur langue d'apprentissage et peuvent avancer à travers différents niveaux qui sont validés par des examens administrés par l'administration.

Chapitre 5

Implémentation

5.1 Processus de Développement et Méthodologies

Le processus de développement suit les méthodologies agiles telles que Scrum, adoptant une approche itérative et incrémentale pour l'implémentation des microservices :

Planification : Après avoir rassemblé les exigences, établi les user stories et mis en place un backlog de produits.

Planification de sprint : Après avoir choisi les user stories pour le backlog de sprint, nous avons déterminé les objectifs à atteindre lors du sprint.

Développement : En utilisant les technologies et frameworks choisis, nous avons mis en place les microservices en suivant les user stories sélectionnées.

Tests : Afin de s'assurer du bon fonctionnement des composants individuels, nous avons procédé à l'exécution de tests unitaires pour chaque microservice. On a également effectué des tests d'intégration afin de vérifier la communication et l'interaction entre les microservices.

Déploiement Chaque microservice a été encapsulé dans des conteneurs à l'aide de Docker, tandis que Kubernetes a été utilisé pour orchestrer et déployer ces conteneurs.

5.2 Détails de l'Implémentation pour Chaque Microservice

MICROSERVICE-AUTH : Le microservice AUTH-MICROSERVICE est implémenté à l'aide de Nest.js et MongoDB, fournissant ainsi un cadre robuste pour développer les fonctionnalités d'authentification et de gestion des sessions. Pour renforcer ses capacités, nous avons intégré des bibliothèques spécialisées dans l'authentification et l'autorisation. Par exemple, le module @nestjs/jwt nous a permis de générer et de valider des jetons JWT (JSON Web Tokens) pour authentifier les utilisateurs et sécuriser les échanges entre le client et le serveur. Cette bibliothèque offre une intégration fluide avec Nest.js et propose des fonctionnalités avancées pour manipuler les jetons JWT en toute sécurité.

En parallèle, nous avons utilisé la bibliothèque bcrypt pour sécuriser le hachage des mots de passe des utilisateurs. Ses fonctions de hachage robustes garantissent la sécurité des mots de passe stockés dans notre base de données, les protégeant ainsi contre les attaques par force brute et les attaques par dictionnaire.

En utilisant conjointement ces bibliothèques et les fonctionnalités natives de Nest. Grâce à l'utilisation de js et MongoDB, notre microservice AUTH-MICROSERVICE présente une plus grande résistance, sécurité et possibilité d'extension. Grâce à ces éléments, notre plateforme d'enseignement en ligne garantit une expérience utilisateur sécurisée et sans faille, renforçant ainsi sa fiabilité et sa sécurité globale.

MICROSERVICE-Groupe : Le microservice MICROSERVICE-Groupe est développé en utilisant Nest.js, et MongoDB, une base de données NoSQL flexible et performante, est utilisée pour stocker les données relatives aux opérations réalisées par ce microservice, telles que les groupes ou sessions créés, les étudiants inscrits dans ces groupes, et autres métadonnées associées. Ce microservice gère plusieurs fonctionnalités essentielles, notamment la création et la gestion des groupes ou sessions par les enseignants, l'inscription des étudiants aux groupes disponibles, la gestion des relations entre enseignants et étudiants, et le suivi des progrès et des performances des étudiants. Les enseignants peuvent créer des groupes en spécifiant des détails tels que le nom du groupe, la date de début, la date limite d'inscription et le nombre maximum d'étudiants. Les étudiants peuvent s'inscrire aux groupes en fournissant leurs informations personnelles, et le microservice met à jour la base de données en conséquence. En combinant les capacités de Nest.js pour la création d'API RESTful et la flexibilité de MongoDB pour la gestion des données, le microservice MICROSERVICE-Groupe offre une solution puissante et extensible pour gérer les groupes et les sessions au sein de notre plateforme d'enseignement en ligne, assurant une évolutivité facile et une adaptation aux besoins futurs.

MICROSERVICE-Paiment : Le microservice MICROSERVICE-Paiement est basé sur Spring Boot et MongoDB.Il est conçu pour gérer toutes les transactions financières de la plateforme. Spring Boot, avec sa structure modulaire et ses

fonctionnalités de sécurité intégrées, permet de développer des API RESTful sécurisées et maintenables. MongoDB, en tant que base de données NoSQL, offre la flexibilité nécessaire pour stocker et organiser efficacement les données de paiement, y compris les les informations des utilisateurs.

Micoservice-Notification : Le microservice MICROSERVICE-Notification, basé sur NestJS et MongoDB, est conçu pour gérer les notifications au sein de la plateforme. MongoDB permet de stocker efficacement les données non structurées. Ce microservice gère l'envoi de notifications pour divers événements, tels que la création de groupes et de sessions, ainsi que l'inscription des étudiants et enseignants,le dépot des documents par les enseignants . Cette approche garantit un système de notification robuste et évolutif pour la plateforme d'enseignement en ligne.

MICROSERVICE-Filtrage : Le microservice MICROSERVICE-Filtrage, développé avec NestJS et MongoDB, joue un rôle important dans la gestion de la filtration des données sur la plateforme. En utilisant NestJS, qui simplifie l'implémentation de fonctionnalités de filtrage. MongoDB, une base de données NoSQL, est utilisée pour stocker et manipuler des volumes importants de données non structurées, offrant ainsi une flexibilité et une performance optimales.

Ce microservice permet aux étudiants de filtrer les informations en fonction de critères tels que l'année, la spécialité, le niveau, la matière et même le prix des cours de soutien. Grâce à cette fonctionnalité de filtrage, les sessions qui correspondent aux critères de recherche de l'étudiant sont rapidement identifiées et affichées. Grâce à NestJS, il peut gérer efficacement des requêtes complexes, tandis que MongoDB permet une recherche rapide et efficace à travers de vastes ensembles de données.

MICROSERVICE Forummessagerie : Le microservice FORUM-MESSAGERIE utilise Node.js et MongoDB pour stocker et gérer les messages échangés entre les utilisateurs de la plateforme. MongoDB offre une flexibilité dans le stockage des données, tandis que Node.js permet de développer des applications évolutives et hautement performantes. De plus, le microservice intègre Socket.IO pour la gestion des communications en temps réel, ce qui garantit une expérience de messagerie fluide et réactive pour les utilisateurs. Avec Socket.IO, les messages sont livrés instantanément, offrant ainsi une interaction en temps réel sans délai perceptible.

MICROSERVICE-Apprentissagelangue : Le MICROSERVICE-ApprentissageLangue est mis en œuvre avec Spring Boot et MongoDB pour offrir une fonctionnalité d'apprentissage de langues sur notre plateforme. L'étudiant peut accéder à cette fonctionnalité et choisir la langue qu'il souhaite apprendre parmi les options disponibles telles que l'arabe, l'anglais ou le français. De plus, il a la possibilité de sélectionner s'il veut se concentrer sur la grammaire ou le vocabulaire. Le processus d'apprentissage se déroule en plusieurs niveaux, chacun comportant plusieurs étapes contenant des informations adaptées

à ce niveau spécifique. Pour progresser d'un niveau à un autre, l'étudiant doit réussir un examen qui sera évalué et validé par l'administrateur. Cette approche structurée et progressive assure un apprentissage efficace et personnalisé des langues pour les utilisateurs de notre plateforme.

FIGURE 5.1 – Nest Logo

FIGURE 5.2 – Spring Logo

Figure 5.3 – MongoDB Logo

Figure 5.4 – NodeJS Logo

5.3 Intégration et Communication entre les Microservices

L'intégration et la communication entre les microservices ont été réalisées en utilisant les approches suivantes :

5.3.1 APIs RESTful :

Pour faciliter la communication et l'interaction entre les différents microservices de notre architecture, nous avons adopté une approche basée sur des API RESTful. Chaque microservice expose des points d'entrée bien définis via des API REST, permettant aux autres composants du système d'interagir avec lui de manière standardisée.

Grâce à cette architecture, les microservices peuvent communiquer entre eux en effectuant des requêtes HTTP aux API exposées. Cela permet une intégration fluide et une collaboration efficace entre les différents modules de notre plateforme. Par exemple, lorsque le MICROSERVICE-AUTH a besoin de vérifier les autorisations d'un utilisateur, il peut envoyer une requête HTTP au MICROSERVICE-PROFIL pour obtenir les informations d'identification de l'utilisateur.

De plus, l'utilisation d'API RESTful offre une flexibilité et une extensibilité accrues à notre architecture. Les microservices peuvent être développés, déployés et mis à l'échelle de manière indépendante, sans perturber le fonctionnement global du système. Cela facilite également la maintenance et l'évolution de chaque composant de manière isolée, ce qui contribue à la robustesse et à la pérennité de notre solution.

5.3.2 Découverte de services :

Pour garantir une communication fluide et dynamique entre les différents microservices de notre architecture, nous avons mis en œuvre un mécanisme de découverte de services en utilisant Netflix Eureka. Cette solution permet à chaque microservice de localiser et de communiquer de manière dynamique avec d'autres microservices, sans avoir à connaître leurs emplacements exacts à l'avance.

Grâce à Netflix Eureka, chaque microservice peut s'inscrire auprès du serveur Eureka lors de son démarrage, ce qui lui permet d'être répertorié dans le registre des services disponibles. Les autres microservices peuvent alors interroger le serveur Eureka pour découvrir les services dont ils ont besoin, en utilisant des noms symboliques plutôt que des adresses IP ou des URL codées en dur. Cette approche décentralisée et basée sur la configuration permet une communication transparente et dynamique entre les microservices, sans nécessiter de coordination manuelle ou de configuration statique.

En utilisant Netflix Eureka, nous avons pu éviter le codage en dur des points de terminaison de service, ce qui rend notre architecture plus flexible et adaptable aux changements. De plus, cela simplifie le déploiement et la gestion des microservices, car ils peuvent être démarrés et arrêtés de manière indépendante sans perturber le fonctionnement global du système.

FIGURE 5.5 – Eureka Client Logo

5.3.3 Protocoles de communication :

Pour assurer une intégration fluide et une communication efficace entre nos microservices, nous avons opté pour l'utilisation de protocoles de communication standard tels que HTTP/REST. Ces protocoles offrent une compatibilité et une interopérabilité élevées, ce qui facilite l'échange de données entre les différents composants du système.

En adoptant HTTP comme protocole de communication principal, nous avons pu tirer parti de ses fonctionnalités bien établies telles que les méthodes de requête standard (GET, POST, PUT, DELETE), les codes de statut HTTP pour la gestion des erreurs et la structuration des données avec JSON. Cela nous a permis de concevoir des API RESTful pour chaque microservice, définissant clairement les points d'entrée, les ressources disponibles et les opérations supportées.

L'utilisation de REST (Representational State Transfer) comme style architectural pour nos API garantit une conception uniforme et cohérente, facilitant ainsi leur compréhension et leur utilisation par les développeurs. De plus, REST favorise la scalabilité et la flexibilité, ce qui est essentiel dans un environnement distribué où les besoins et les exigences peuvent évoluer avec le temps.

Grâce à l'utilisation de protocoles de communication standard comme HTTP/REST, nous avons pu garantir une intégration facile et une communication efficace entre nos microservices.

5.3.4 Kafka :

En plus des méthodes de communication synchrones basées sur HTTP/REST, nous avons incorporé Apache Kafka pour favoriser la communication asynchrone et événementielle entre nos microservices. Apache Kafka offre une architecture de streaming de données distribuée qui permet le transfert fiable et en temps réel des messages entre les différents composants du système.

L'utilisation de Kafka nous permet de déplacer la communication entre microservices vers un modèle basé sur des événements, où les services peuvent produire et consommer des messages de manière déconnectée. Cela présente plusieurs avantages, notamment une meilleure scalabilité, une résilience accrue et une réactivité améliorée du système dans des scénarios où les charges de travail peuvent être variables et imprévisibles.

Grâce à Kafka, les microservices peuvent communiquer de manière asynchrone sans bloquer le flux principal de traitement, ce qui réduit les dépendances et les temps d'attente entre les services. De plus, Kafka offre des fonctionnalités avancées telles que la réplication, la partitionnement et la gestion des décalages de consommation, ce qui garantit la fiabilité et la cohérence des messages échangés même en cas de défaillance ou de redémarrage des microservices.Les deux microservices qui sont inscrit à kafka sont : Ms-auth et Ms-Groupe.

FIGURE 5.6 – Kafka Logo

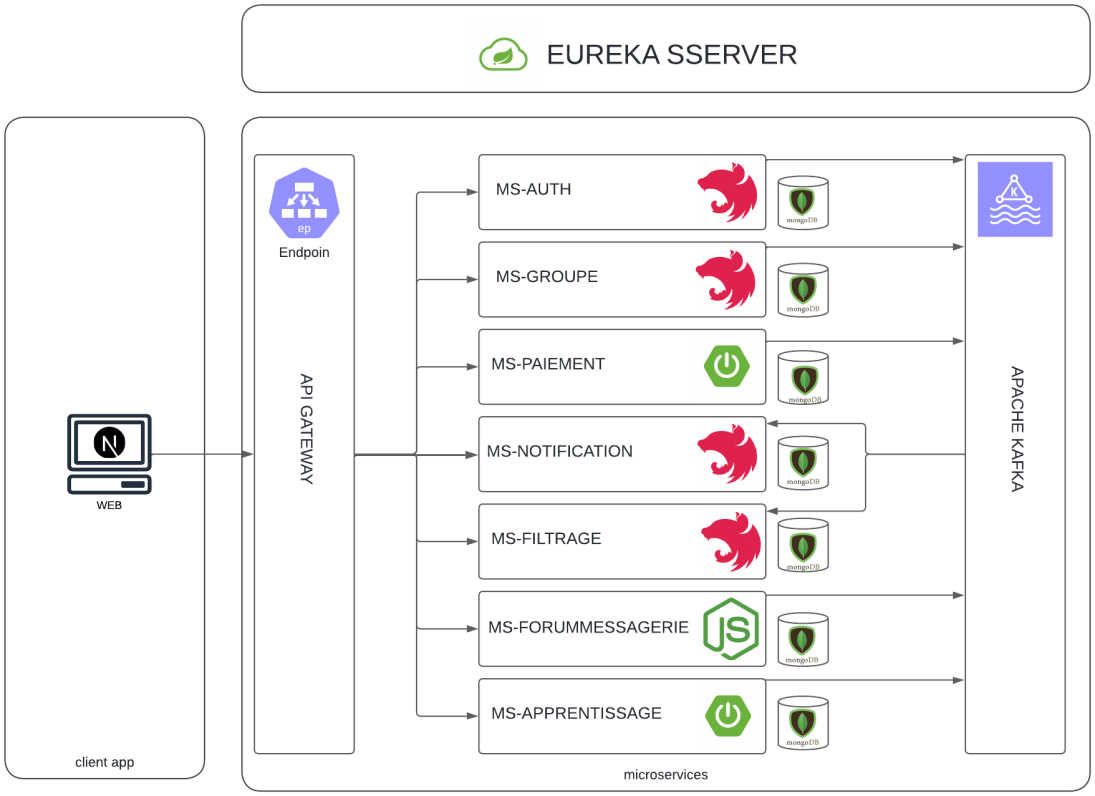

FIGURE 5.7 – Fonctionnement de l'architecture des microservices

5.4 Gestion de données des microservices

Afin de gérer efficacement les données dans notre système de microservices, nous avons adopté une approche où chaque service dispose de sa propre base de données dédiée. Pour orchestrer les opérations de commande entre certains microservices, nous avons mis en place l'architecture CQRS (Command Query Responsibility Segregation).

L'architecture CQRS : L'architecture CQRS nous permet de séparer les opérations de lecture (query) et d'écriture (command) sur les données. Cette séparation améliore la performance, la scalabilité et la maintenabilité du système, en permettant une gestion plus fine des accès et des modifications des données.

Dans notre cas,les microservices de commande (ms-command-auth et ms-command-group) reçoivent des commandes via des API REST, qui sont ensuite traitées par des gateways de commande, envoyées via un bus de commande et gérées par des handlers qui mettent à jour le modèle de domaine (prof, student, user, session, group). Les modifications sont enregistrées dans une base de données d'écriture et un store d'événements pour une traçabilité complète. Les microservices de requête (ms-Query-Filtrage et ms-Query-Notification) reçoivent des requêtes via des API REST, transmises par des gateways de requête à travers un bus de requête et traitées par des handlers qui interrogent des bases de données MongoDB pour obtenir les données nécessaires (prof, session, notification). Cette séparation des préoccupations de lecture et d'écriture améliore la performance, la scalabilité et la traçabilité du système.

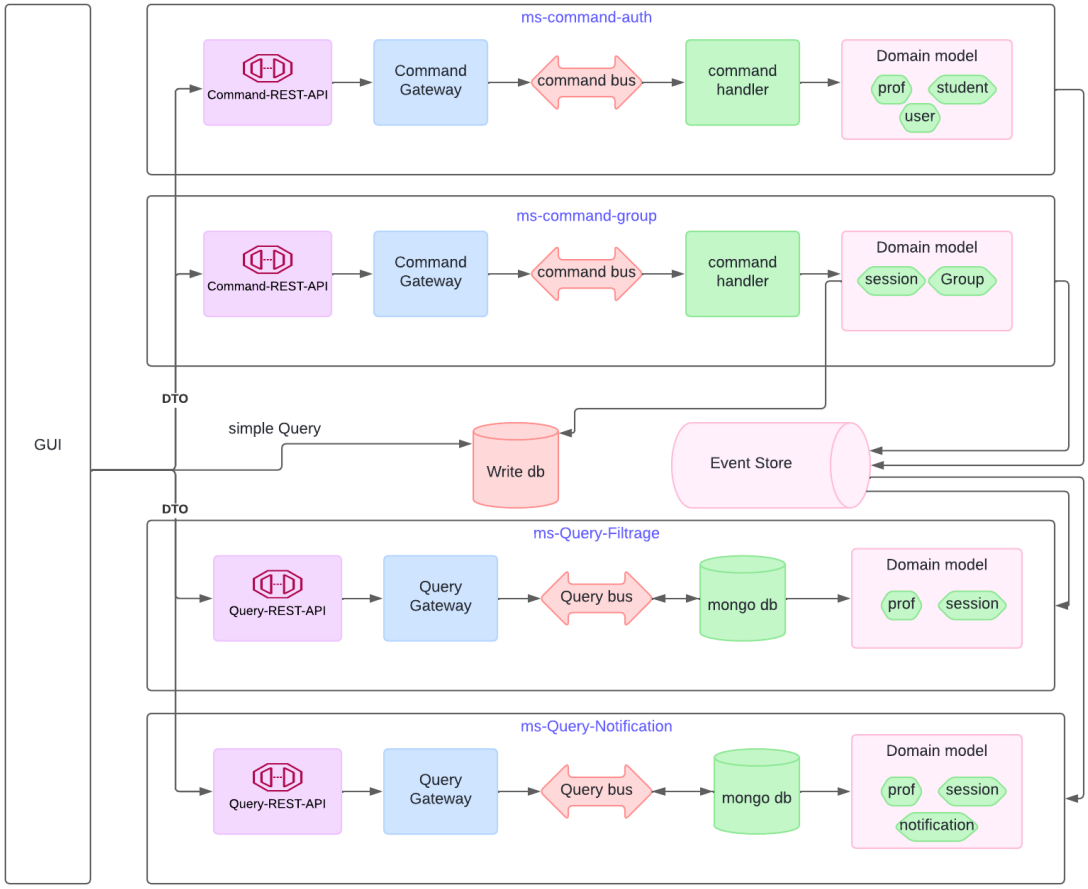

FIGURE 5.8 – Architecture CQRS

5.5 Implémentation d'Interface Utilisateur

5.5.1 Next JS :

Pour le développement frontend, nous avons utilisé Next.js, un framework gratuit et open source s'appuyant sur la bibliothèque JavaScript React et sur la technologie Node.js. Avec Next.js, nous avons pu structurer notre code de manière modulaire, en utilisant des composants réutilisables tels que les directives, les contrôleurs et les services. Cela nous a permis de développer l'interface utilisateur de manière efficace et de maintenir une organisation claire du code.

Next.js offre également des fonctionnalités puissantes telles que le rendu côté serveur (SSR) et le rendu statique (SSG), qui améliorent les performances et l'optimisation pour les moteurs de recherche (SEO). De plus, Next.js propose un système de routage intégré, qui permet de gérer la navigation entre les différentes vues de l'application sans rechargement de la page. Cela offre une expérience utilisateur fluide et permet une navigation transparente au sein de l'application.

En utilisant Next.js, nous avons pu créer une interface utilisateur réactive, conviviale et hautement interactive pour notre plateforme. Nous avons utilisé les fonctionnalités avancées de React pour gérer la logique de présentation, les formulaires, les interactions utilisateur et bien plus encore.

Ona pu bnéficier des nombreuses bibliothèques comme par exemple NextAuth.js v14 pour la gestion de l'authentification des utilisateurs, nous avons intégré NextAuth.js version 14. NextAuth.js est une solution d'authentification flexible et sécurisée pour les applications Next.js. Il prend en charge plusieurs fournisseurs d'authentification, tels que Google, Facebook, et bien d'autres, facilitant ainsi l'intégration de l'authentification sociale.

Avec NextAuth.js v14, nous avons pu mettre en place une authentification robuste avec des sessions sécurisées et une gestion simplifiée des utilisateurs. L'intégration de cette bibliothèque dans notre projet Next.js nous a permis de gérer efficacement les connexions des utilisateurs, les autorisations et les sessions.

FIGURE 5.9 – Next js Logo

Chapitre 6

Conclusion

Notre projet se concentre sur le développement d'une plateforme d'externalisation basée sur une architecture de microservices. L'objectif principal de la plateforme est de faciliter la mise en relation entre les travailleurs indépendants et les entreprises ayant des besoins spécifiques. Cela permet aux travailleurs de proposer leurs compétences et disponibilités, tandis que les entreprises peuvent publier des offres d'embauche et recruter des travailleurs correspondant à leurs besoins.

L'architecture de microservices a été choisie en raison de ses avantages en termes de flexibilité, de scalabilité et de gestion efficace de composants autonomes. Chaque microservice est responsable d'une fonctionnalité spécifique, telle que l'authentification, la gestion des interactions, les paiements, le stockage des fichiers, la gestion des travailleurs et la gestion des entreprises. Cela permet une conception modulaire et facilite la maintenance et l'évolutivité du système.

Pour mettre en œuvre notre projet, nous avons utilisé différentes technologies et frameworks adaptés à chaque microservice. Par exemple, le microservice d'authentification, d'interaction et de paiement a été développé en utilisant Spring Boot et MySQL. Le microservice de stockage des fichiers a également utilisé Spring Boot et MySQL. Pour le microservice de gestion des travailleurs, nous avons utilisé Spring Boot et MongoDB, tandis que le microservice de gestion des entreprises a utilisé Spring Boot et MongoDB.

L'intégration et la communication entre les microservices sont essentielles pour assurer le bon fonctionnement de notre système. Nous avons mis en place des API RESTful pour permettre une communication fluide entre les microservices. Nous avons également utilisé des mécanismes de découverte de services tels que Netflix Eureka pour faciliter la localisation et la communication dynamique entre les microservices.

Au cours du processus de développement, nous avons suivi des méthodologies agiles telles que Scrum, ce qui nous a permis de livrer des fonctionnalités de manière itérative et d'incorporer régulièrement les commentaires des parties prenantes. Nous avons accordé une attention particulière aux tests automatisés, à la conteneurisation avec Docker et au déploiement avec Kubernetes pour garantir une mise en production efficace et fiable.

En conclusion, notre projet de plateforme d'externalisation basée sur une

architecture de microservices offre une solution flexible, évolutive et modulaire pour la gestion des travailleurs indépendants et des besoins des entreprises. Les différentes technologies utilisées et les approches d'intégration adoptées garantissent une communication fluide et une évolutivité du système, répondant ainsi aux exigences du marché de l'externalisation et favorisant la collaboration efficace entre les travailleurs et les entreprises.