

DOCUMENTATION DU PROJET DÉVELOPPER ET SÉCURISER LES MICRO-SERVICES

ABDESSADEQ MAKKIOUI
FATIMA ZZAHRA KAIOU
MARIAM LAGHFIRI
OMAYMA ELBAKRI
YASSIN EZAOUIBI

Micro Service :

Un microservice est une approche de développement logiciel où une application est divisée en plusieurs petits services indépendants, chacun réalisant une tâche spécifique. Ces services sont déployés séparément et communiquent entre eux via des protocoles légers comme HTTP ou RPC(Remote Procedure Call). Cette architecture favorise la modularité, la scalabilité et la flexibilité des applications.

Service Configuration :

Un service de configuration dans un projet de microservices est un composant qui gère la configuration des différents microservices de l'application. Il fournit un moyen centralisé pour stocker et distribuer les paramètres de configuration nécessaires à chaque microservice. Cela inclut souvent des informations telles que les URL des bases de données, les clés d'API, les paramètres de sécurité, etc

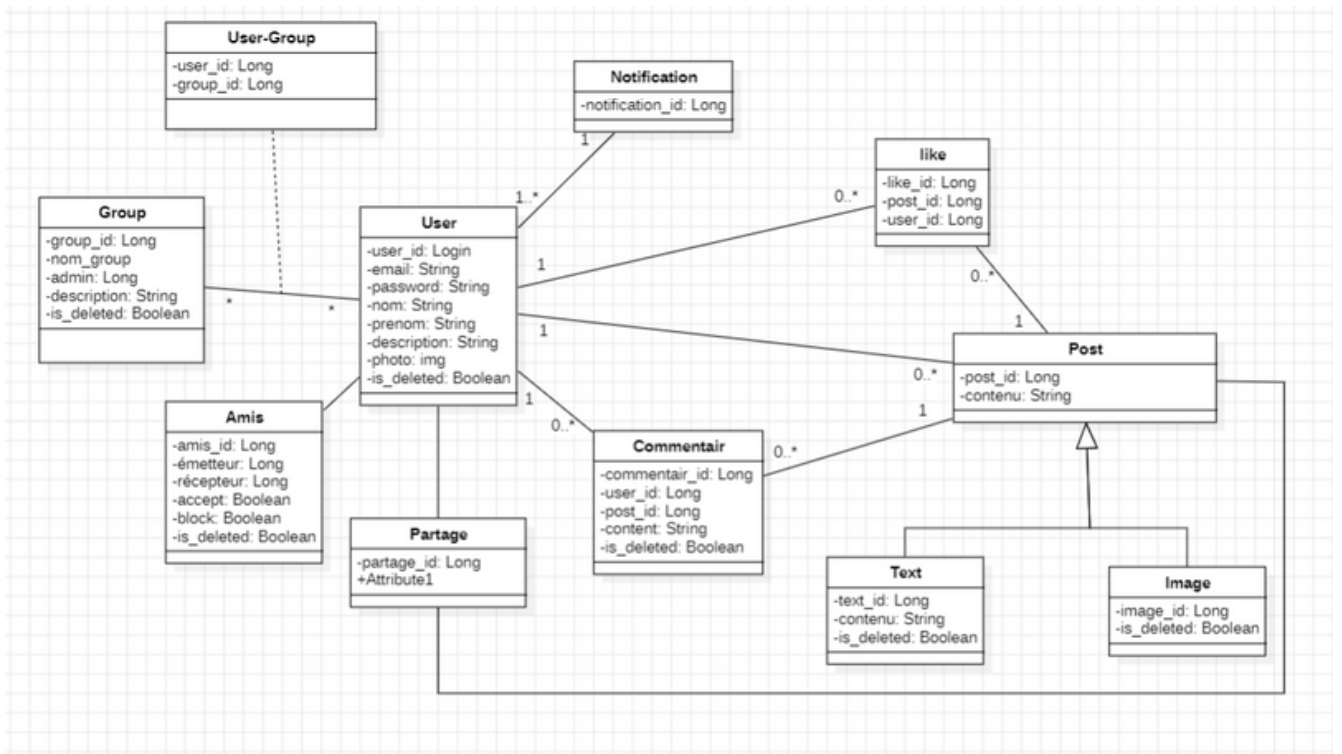
Service Gateway :

Un service de passerelle (Gateway) dans un projet de microservices est un point d'entrée centralisé pour les clients externes, routant les requêtes vers les microservices appropriés en fonction de critères comme l'URL ou les en-têtes HTTP. Il gère également des fonctionnalités telles que l'authentification, l'autorisation, la sécurité et la transformation des données, simplifiant ainsi l'accès aux microservices tout en fournissant des fonctionnalités de gestion centralisées.

Service Discovery :

Le Service Discovery est un composant essentiel dans l'architecture des microservices. Il permet aux différents services d'un système distribué de découvrir et de communiquer entre eux de manière dynamique, sans nécessiter de configuration statique. En résumé, le Service Discovery offre un mécanisme automatique pour localiser et interagir avec les services disponibles dans l'environnement, facilitant ainsi le déploiement, la scalabilité et la résilience des applications distribuées.

Diagramme de Classe :

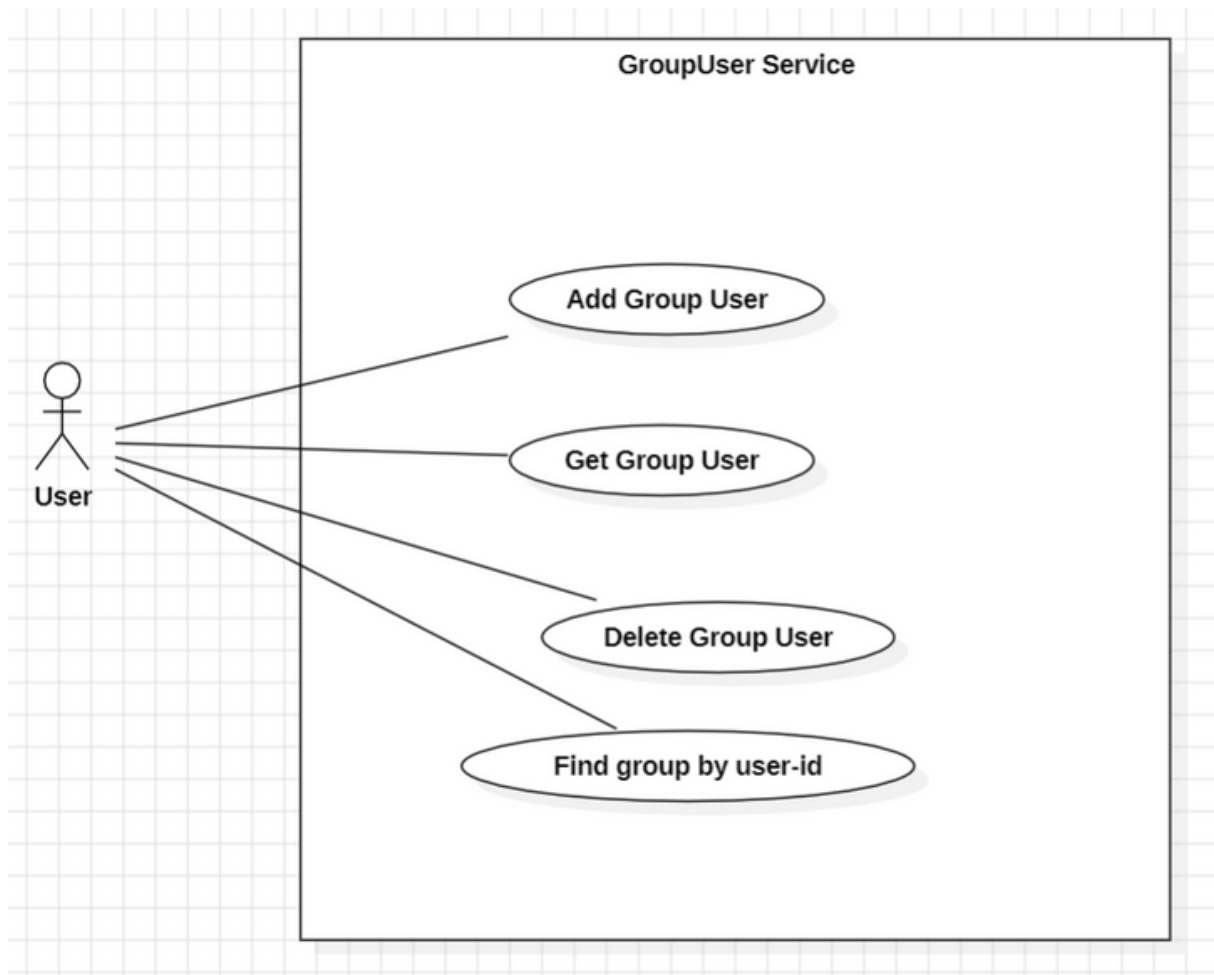


SERVICE GROUP-USER

Introduction :

Ce micro-service est conçu pour faciliter la gestion des interactions entre les entités "Utilisateur" et "Groupe", en permettant aux utilisateurs de rejoindre, quitter et interagir avec différents groupes au sein de notre plateforme.

Diagramme Cas D'Utilisation



Fonctionnalités

Afficher tous les utilisateurs de groupe (showAllGroupUsers):

Cette fonctionnalité permet de récupérer la liste de tous les utilisateurs de groupe enregistrés dans le système.

GET http://localhost:4441/group-user/all Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
-----	-------	-------------	-----	-----------

body Cookies Headers (5) Test Results 200 OK 64 ms 405 B Save as example

Pretty Raw Preview Visualize JSON

```
2 {
3   "id": 3,
4   "groupId": 1,
5   "userId": 1,
6   "deleted": false
7 },
8 {
9   "id": 4,
10  "groupId": 1,
11  "userId": 2,
12  "deleted": false
13 },
14 {
15   "id": 6,
16   "groupId": 2,
```

Trouver les groupes par identifiant d'utilisateur (findGroupByUserId):
Cette fonctionnalité permet de rechercher et de récupérer la liste des groupes auxquels un utilisateur spécifique est membre, en utilisant son identifiant d'utilisateur.

GET http://localhost:4441/group-user/5 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
-----	-------	-------------	-----	-----------

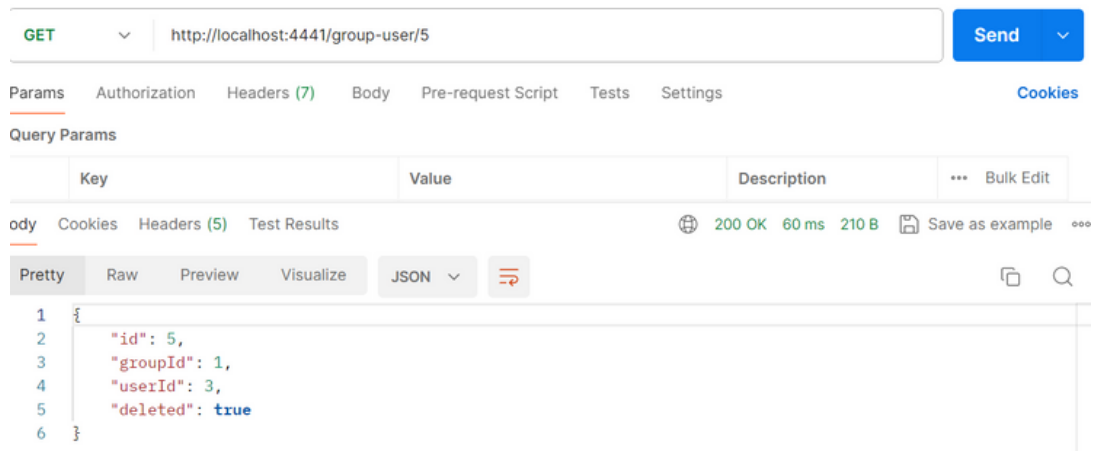
body Cookies Headers (5) Test Results 200 OK 60 ms 210 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 5,
3   "groupId": 1,
4   "userId": 3,
5   "deleted": true
6 }
```

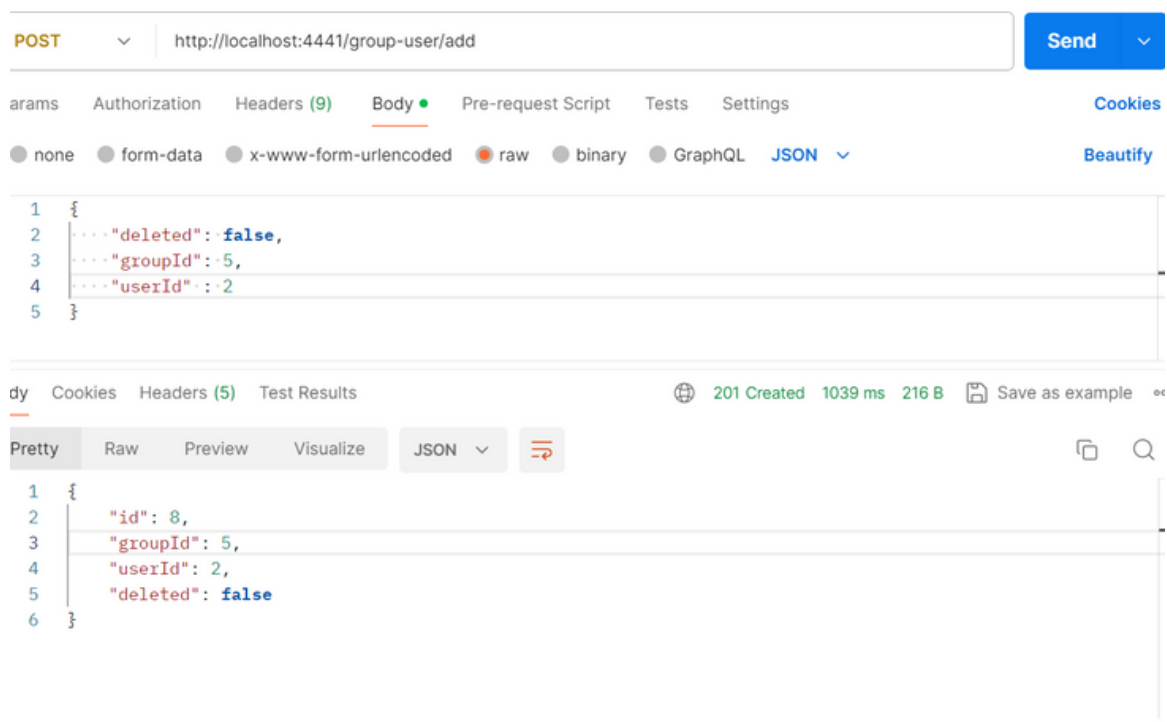
Obtenir un utilisateur de groupe par identifiant (getGroupUser):

Cette fonctionnalité permet de récupérer les détails d'un utilisateur de groupe spécifique en utilisant son identifiant.



Ajouter un utilisateur à un groupe (addGroupUser):

Cette fonctionnalité permet d'ajouter un nouvel utilisateur à un groupe existant en fournissant les détails de l'utilisateur de groupe à ajouter.



Supprimer un utilisateur de groupe par identifiant (deleteGroupUser):

Cette fonctionnalité permet de supprimer un utilisateur de groupe spécifique en utilisant son identifiant.

The screenshot displays a REST client interface. At the top, a request is configured with the method **DELETE** and the URL `http://localhost:4441/group-user/delete/5`. Below the URL bar, tabs for **Params**, **Authorization**, **Headers (7)**, **Body**, **Pre-request Script**, **Tests**, and **Settings** are visible. The **Query Params** section contains a table with columns **Key**, **Value**, and **Description**.

Key	Value	Description
Key	Value	Description

Below the query params, the **Body** tab is selected, showing a JSON response in **Pretty** format. The response is `{ "delete": true }`. The status bar at the top right indicates a **200 OK** status with a response time of **32 ms** and a body size of **179 B**. A **Save as example** button is also present.

```
1 {  
2   "delete": true  
3 }
```

Service de Partage

Le service de partage que vous avez développé semble être une partie d'une application qui permet aux utilisateurs de partager des posts avec d'autres utilisateurs. Voici une documentation de haut niveau sur le service de partage :

Description du Service de Partage

Le service de partage permet aux utilisateurs de partager des posts avec d'autres utilisateurs de l'application. Voici un aperçu des fonctionnalités offertes par ce service :

1. Récupération des Partages

- Le service permet de récupérer la liste de tous les partages effectués par les utilisateurs.

2. Récupération d'un Partage par ID

- Il offre la possibilité de récupérer un partage spécifique en fonction de son identifiant.

3. Enregistrement d'un Partage

- Les utilisateurs peuvent créer et enregistrer de nouveaux partages dans le système. Lorsqu'un utilisateur partage un post, il est enregistré dans la base de données.

4. Suppression d'un Partage

- Les utilisateurs ont la possibilité de supprimer un partage existant à l'aide de son identifiant.

5. Notifications de Partage

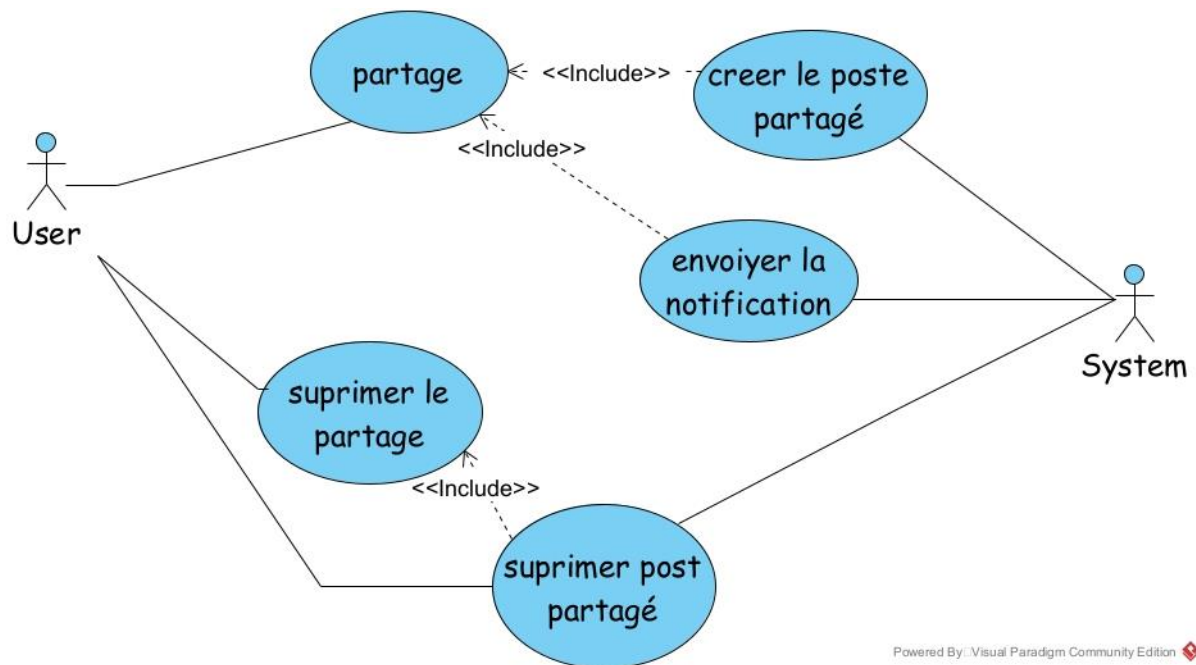
- Lorsqu'un utilisateur partage un post, le service génère une notification pour informer les autres utilisateurs de ce partage.

Fonctionnalités Clés

- **Gestion des Partages:** Le service permet une gestion complète des partages, y compris la création, la récupération et la suppression des partages.
- **Intégration avec d'autres Services :** Le service de partage interagit avec d'autres services tels que le service d'authentification, le service de notification et le service de publication de posts.
- **Mapping des Données :** Le service utilise un système de mapping pour convertir les objets entre les entités métier et les objets de transfert de données (DTO).
- **Intégration avec des Clients Feign :** Le service communique avec d'autres microservices via des clients Feign pour effectuer des opérations telles que la récupération d'utilisateurs, la publication de notifications et la gestion des posts.

Technologies Utilisées

- **Spring Framework:** Le service est construit en utilisant le framework Spring, ce qui lui permet de bénéficier des fonctionnalités telles que l'injection de dépendances et la gestion des requêtes HTTP.
- **Feign Client:** Le service utilise Feign pour simplifier l'appel aux API REST d'autres microservices.



Service Commentaire

Ce service gère la création, la récupération, la mise à jour et la suppression de commentaires.

Endpoints :

1. GET /commentaires/{idPost}

- Description : Récupère tous les commentaires associés à un post spécifique.
- Paramètres de requête :
 - idPost (Long) : L'identifiant du post pour lequel les commentaires doivent être récupérés.
- Réponse : Liste des commentaires associés au post spécifié.

2. GET /commentaires/{id}

- Description : Récupère un commentaire spécifique par son identifiant.
- Paramètres de requête :
 - id (Long) : L'identifiant du commentaire à récupérer.
- Réponse : Le commentaire spécifié s'il existe, sinon une réponse HTTP NOT FOUND.

3. POST /commentaires

- Description : Crée un nouveau commentaire.
- Corps de la requête : Objet JSON représentant le nouveau commentaire.
- Réponse : Le commentaire créé avec son identifiant.

4. PUT /commentaires/{id}

- Description : Met à jour un commentaire existant.
- Paramètres de requête :
 - id (Long) : L'identifiant du commentaire à mettre à jour.
- Corps de la requête : Objet JSON représentant les modifications à apporter au commentaire.
- Réponse : Le commentaire mis à jour.

5. DELETE /commentaires/{id}

- Description : Supprime un commentaire existant.
- Paramètres de requête :
 - id (Long) : L'identifiant du commentaire à supprimer.

- Réponse : Réponse HTTP NO CONTENT.

Service Interaction

Ce service gère la création, la récupération, la mise à jour et la suppression d'interactions.

Endpoints :

1. GET /interactions

- Description : Récupère toutes les interactions.
- Réponse : Liste de toutes les interactions.

2. GET /interactions/{id}

- Description : Récupère une interaction spécifique par son identifiant.
- Paramètres de requête :
 - id (Long) : L'identifiant de l'interaction à récupérer.
- Réponse : L'interaction spécifiée s'il existe, sinon une réponse HTTP NOT FOUND.

3. GET /interactions/NumberInteraction/{idPost}

- Description : Récupère le nombre total d'interactions pour un post spécifique.
- Paramètres de requête :
 - idPost (Long) : L'identifiant du post pour lequel le nombre d'interactions doit être récupéré.
- Réponse : Le nombre total d'interactions pour le post spécifié.

4. POST /interactions

- Description : Crée une nouvelle interaction.
- Corps de la requête : Objet JSON représentant la nouvelle interaction.
- Réponse : L'interaction créée avec son identifiant.

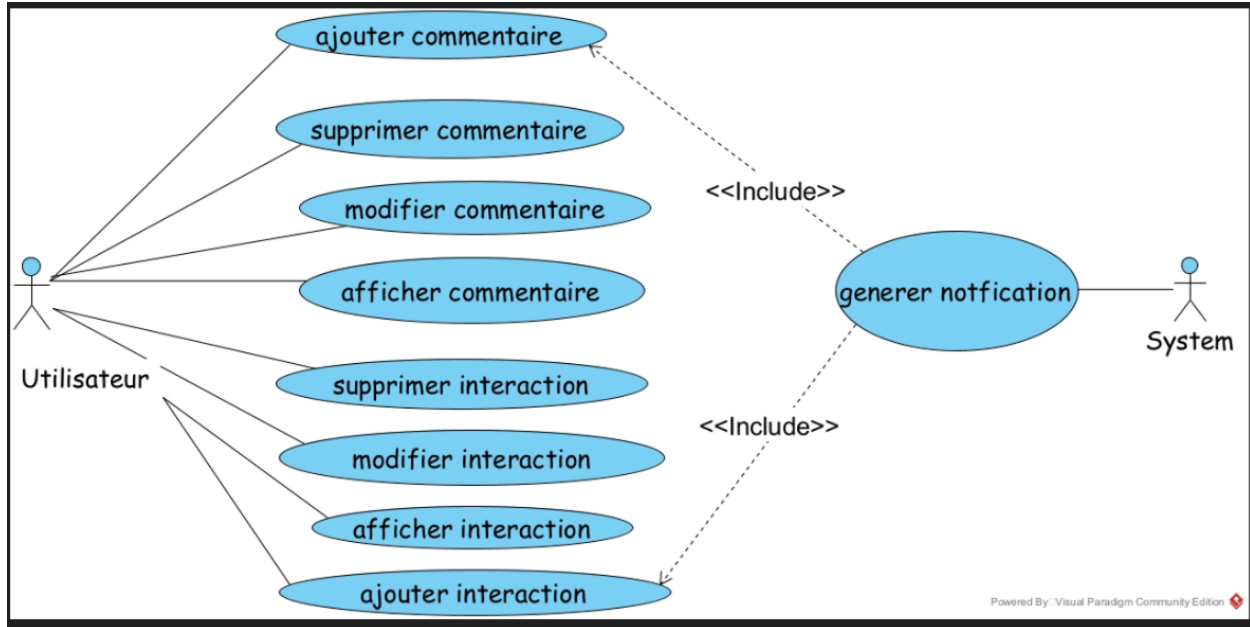
5. PUT /interactions/{id}

- Description : Met à jour une interaction existante.
- Paramètres de requête :
 - id (Long) : L'identifiant de l'interaction à mettre à jour.
- Corps de la requête : Objet JSON représentant les modifications à apporter à l'interaction.
- Réponse : L'interaction mise à jour.

6. DELETE /interactions/{id}

- Description : Supprime une interaction existante.
- Paramètres de requête :
 - id (Long) : L'identifiant de l'interaction à supprimer.
- Réponse : Réponse HTTP NO CONTENT.

c

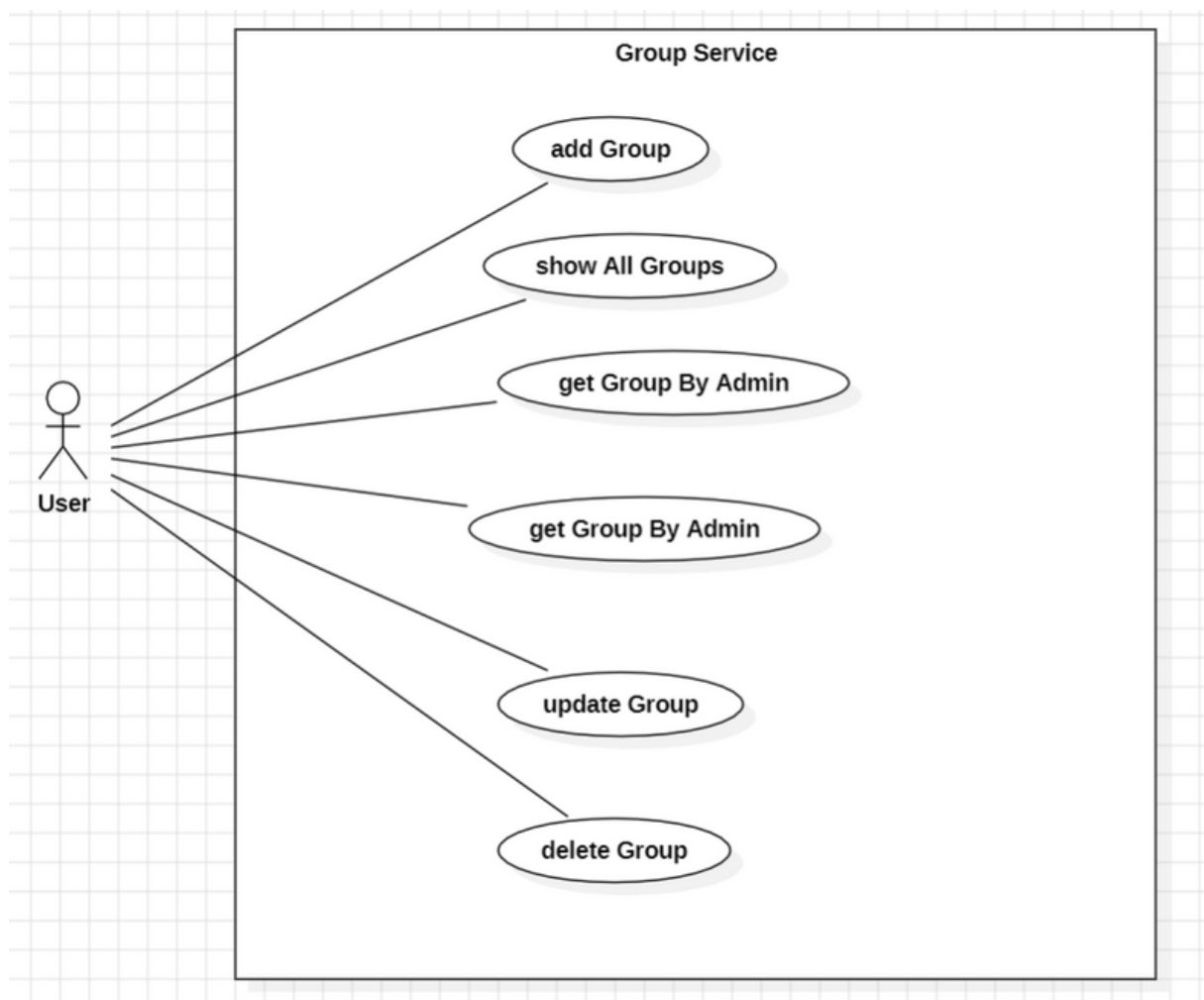


SERVICE GROUP

Introduction :

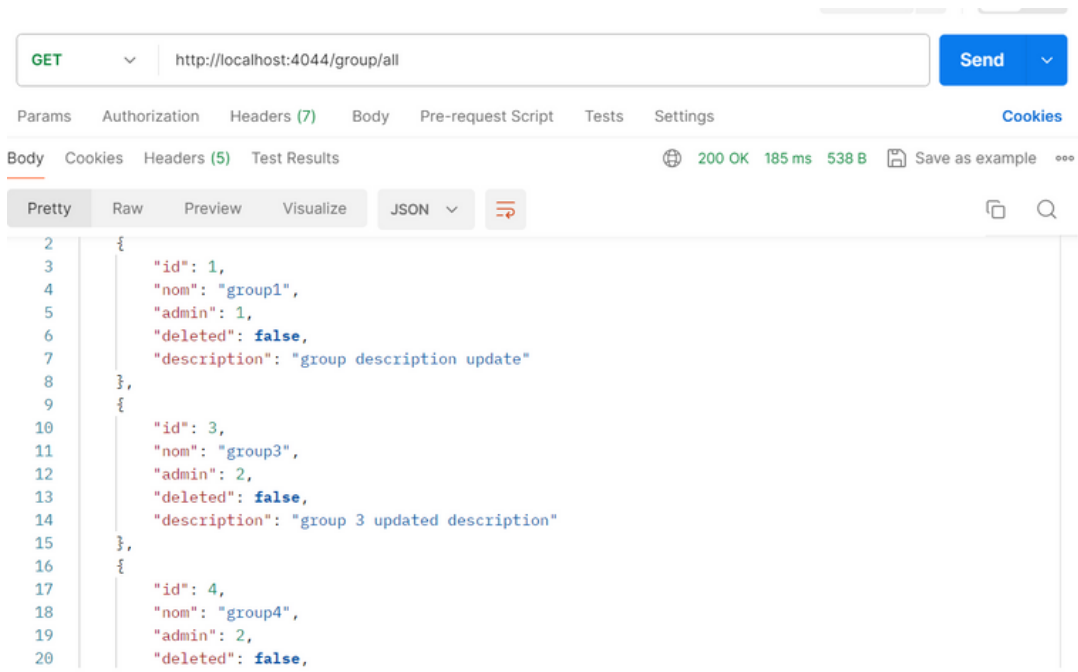
Le micro-service "Group" est une composante essentielle d'une application de réseau social, conçue pour gérer efficacement les interactions au sein des groupes

Diagramme Cas D'Utilisation

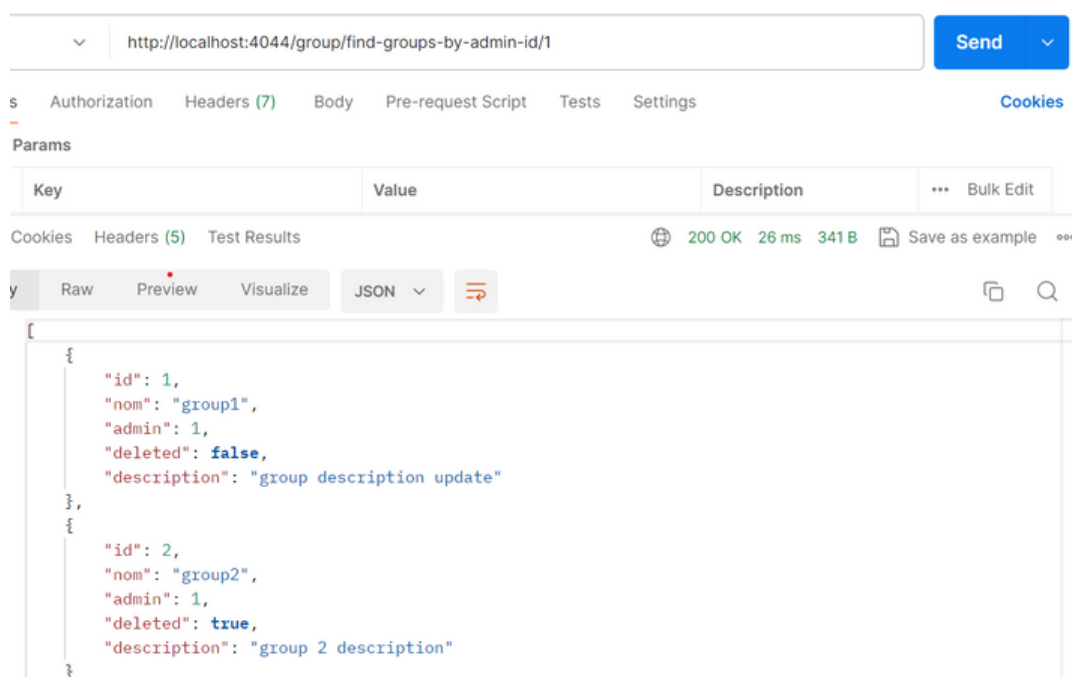


Fonctionnalités

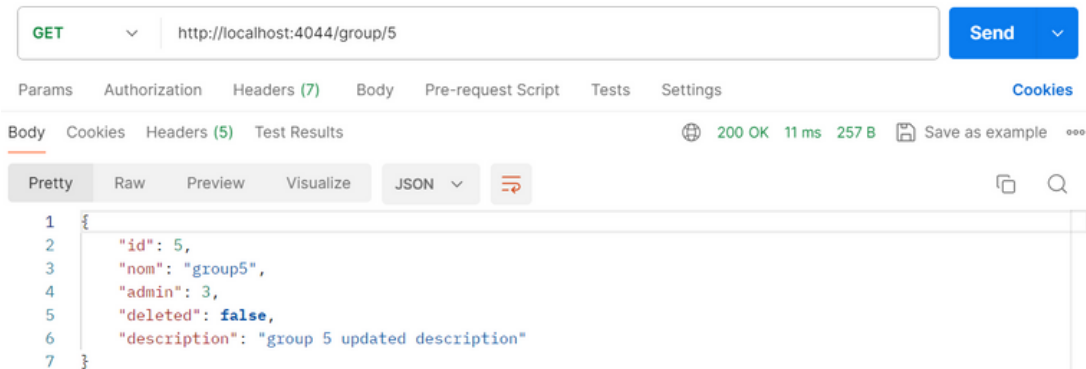
showAllGroups() : Cette méthode est accessible via une requête HTTP GET sur l'endpoint "/group/all". Elle renvoie la liste de tous les groupes présents dans l'application.



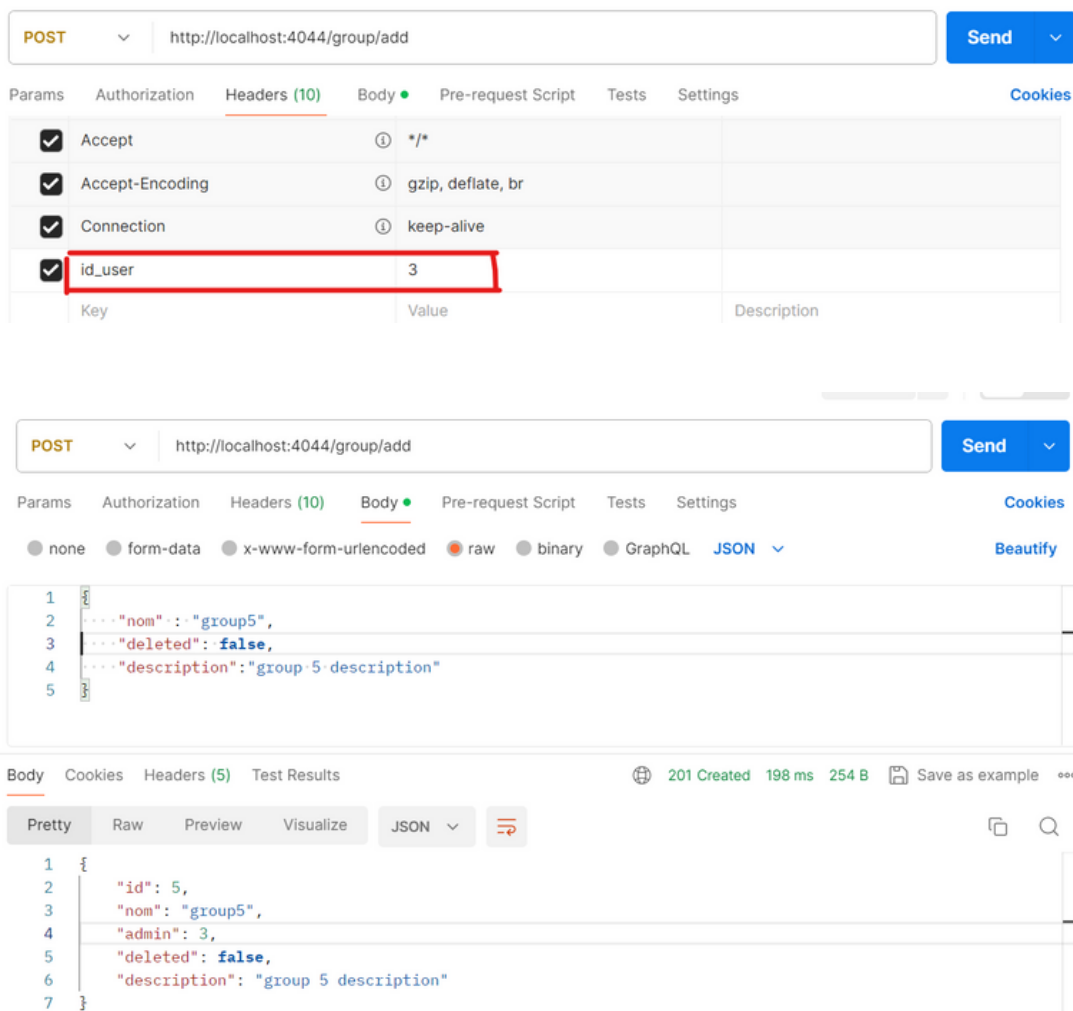
FindGroupByAdmin(Long adminId) : Cette méthode est accessible via une requête HTTP GET sur l'endpoint `"/group/find-groups-by-admin-id/{adminId}"`. Elle permet de trouver tous les groupes gérés par un administrateur spécifique, identifié par son ID.



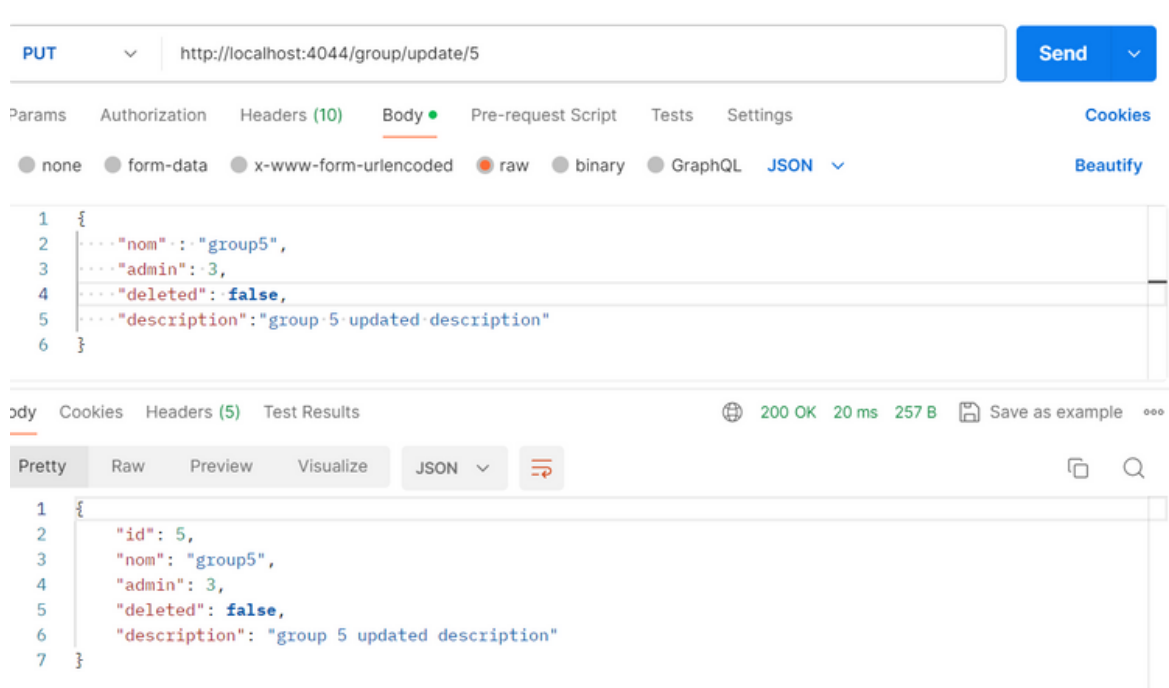
GetGroup(Long group_id) : Cette méthode est accessible via une requête HTTP GET sur l'endpoint `"/group/{id}"`. Elle récupère les détails d'un groupe spécifique en fonction de son ID.



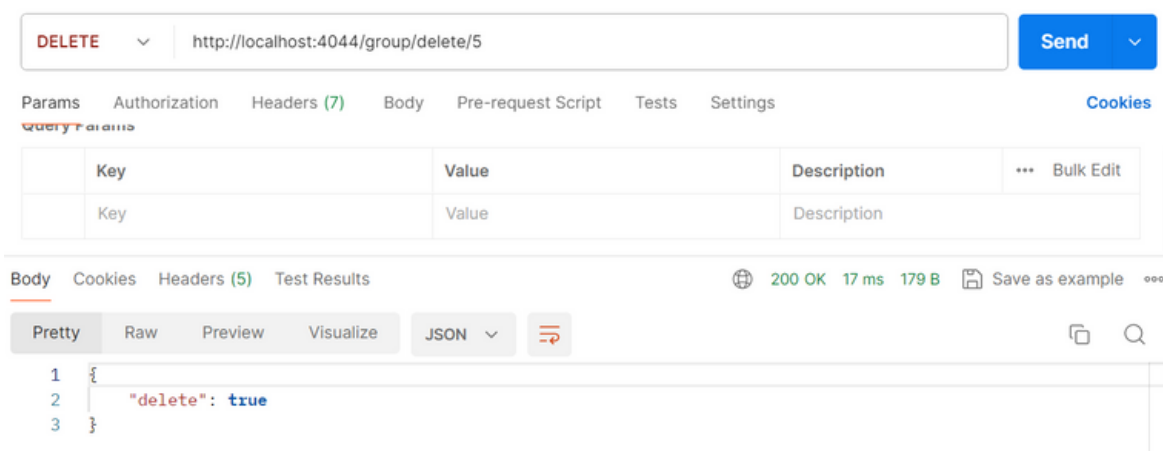
AddGroup(GroupDTO groupDTO, long admin) : Cette méthode est accessible via une requête HTTP POST sur l'endpoint `"/group/add"`. Elle permet d'ajouter un nouveau groupe à l'application, en prenant en compte les informations fournies dans l'objet GroupDTO, et en spécifiant l'administrateur du groupe via l'entête `"id_user"`.



UpdateGroup(GroupDTO groupDTO, Long id) : Cette méthode est accessible via une requête HTTP PUT sur l'endpoint `"/group/update/{id}"`. Elle permet de mettre à jour les informations d'un groupe existant en fonction de son ID, en utilisant les données fournies dans l'objet GroupDTO.



DeleteGroup(Long group_id) : Cette méthode est accessible via une requête HTTP DELETE sur l'endpoint `"/group/delete/{id}"`. Elle permet de supprimer un groupe spécifique en fonction de son ID.



Test

✓ GroupServiceImpTest (org.example.group.service.serviceImp)

- ✓ Delete Group
- ✓ Update Group
- ✓ Get All Groups By Admin Id
- ✓ Add Group
- ✓ Get Group By Id
- ✓ Get All Groups

Documentation des Service Ami & Service Notification

“Micro-Services”

Service d'Amis

Le service d'Amis permet aux utilisateurs de se connecter et d'interagir avec d'autres utilisateurs de l'application en tant qu'amis. Voici un aperçu des fonctionnalités offertes par ce service :

Récupération des Amis

- Le service permet de récupérer la liste de tous les amis d'un utilisateur.

Ajout d'un Ami

- Il offre la possibilité d'ajouter un nouvel ami à la liste d'amis d'un utilisateur.

Suppression d'un Ami

- Les utilisateurs ont la possibilité de supprimer un ami existant de leur liste d'amis.

Notifications d'Amis

- Lorsqu'un utilisateur ajoute un nouvel ami, le service génère une notification pour informer les deux utilisateurs concernés.

Fonctionnalités Clés

- Gestion des Amis : Le service permet une gestion complète des amis, y compris l'ajout, la récupération et la suppression des amis.

- Intégration avec d'autres Services : Le service d'Amis interagit avec d'autres services tels que le service d'authentification, le service de notification et d'autres services de réseau social.

- Mapping des Données : Le service utilise un système de mapping pour convertir les objets entre les entités métier et les objets de transfert de données (DTO).

- Intégration avec des Clients Feign : Le service communique avec d'autres microservices via des clients Feign pour effectuer des

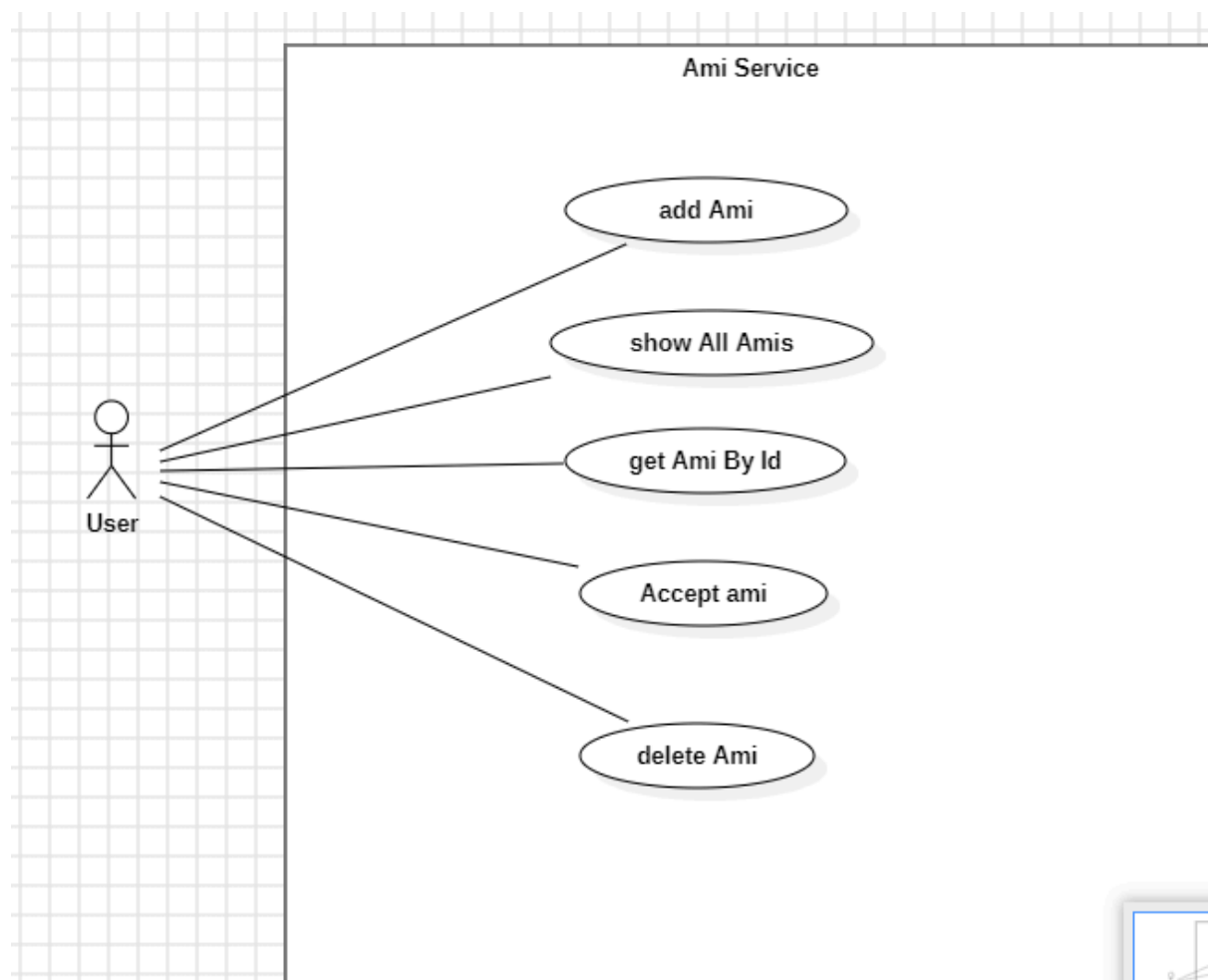
opérations telles que la récupération d'utilisateurs et la gestion des notifications.

Technologies Utilisées

- Spring Framework : Le service est construit en utilisant le framework Spring, ce qui lui permet de bénéficier des fonctionnalités telles que l'injection de dépendances et la gestion des requêtes HTTP.
- Feign Client : Le service utilise Feign pour simplifier l'appel aux API REST d'autres microservices.

Diagramme de Cas d'Utilisation:

Le diagramme de cas d'utilisation ci-dessous illustre les interactions entre les acteurs et le système du service d'Amis :



Service de Notification:

Le service de Notification permet à l'application d'informer les utilisateurs sur les événements importants ou les actions effectuées dans le système. Cette documentation fournit une vue d'ensemble des fonctionnalités, des cas d'utilisation et des technologies utilisées dans le service de Notification.

Description du Service

Le service de Notification gère la création, la récupération et la suppression des notifications pour les utilisateurs de l'application. Il offre des fonctionnalités pour ajouter et supprimer des notifications, ainsi que pour les envoyer aux utilisateurs concernés.

Fonctionnalités Offertes:

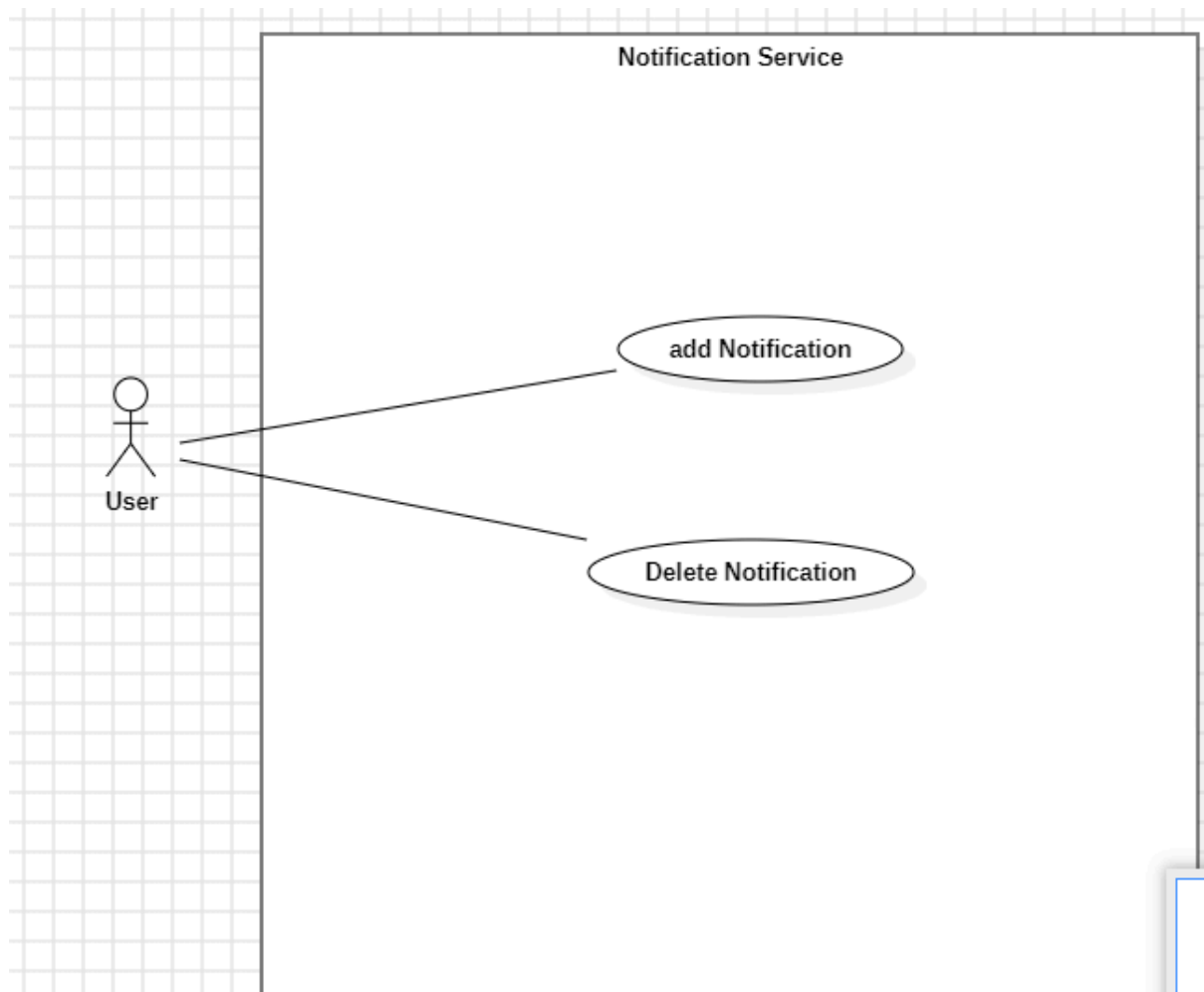
Ajout de Notification :

Permet d'ajouter une nouvelle notification dans le système.

Suppression de Notification :

Permet de supprimer une notification existante du système.

Diagramme de cas d'utilisation:



Technologies Utilisées:

Spring Framework : Utilisé pour la construction du service, ce qui permet d'exploiter des fonctionnalités telles que l'injection de dépendances et la gestion des requêtes HTTP.

Base de Données : Utilisée pour stocker les notifications créées par les utilisateurs.

Intégration avec d'Autres Services:

Le service de Notification est intégré avec d'autres services de l'application, notamment le service d'authentification pour l'identification des utilisateurs et le service de gestion des utilisateurs pour l'envoi ciblé des notifications.

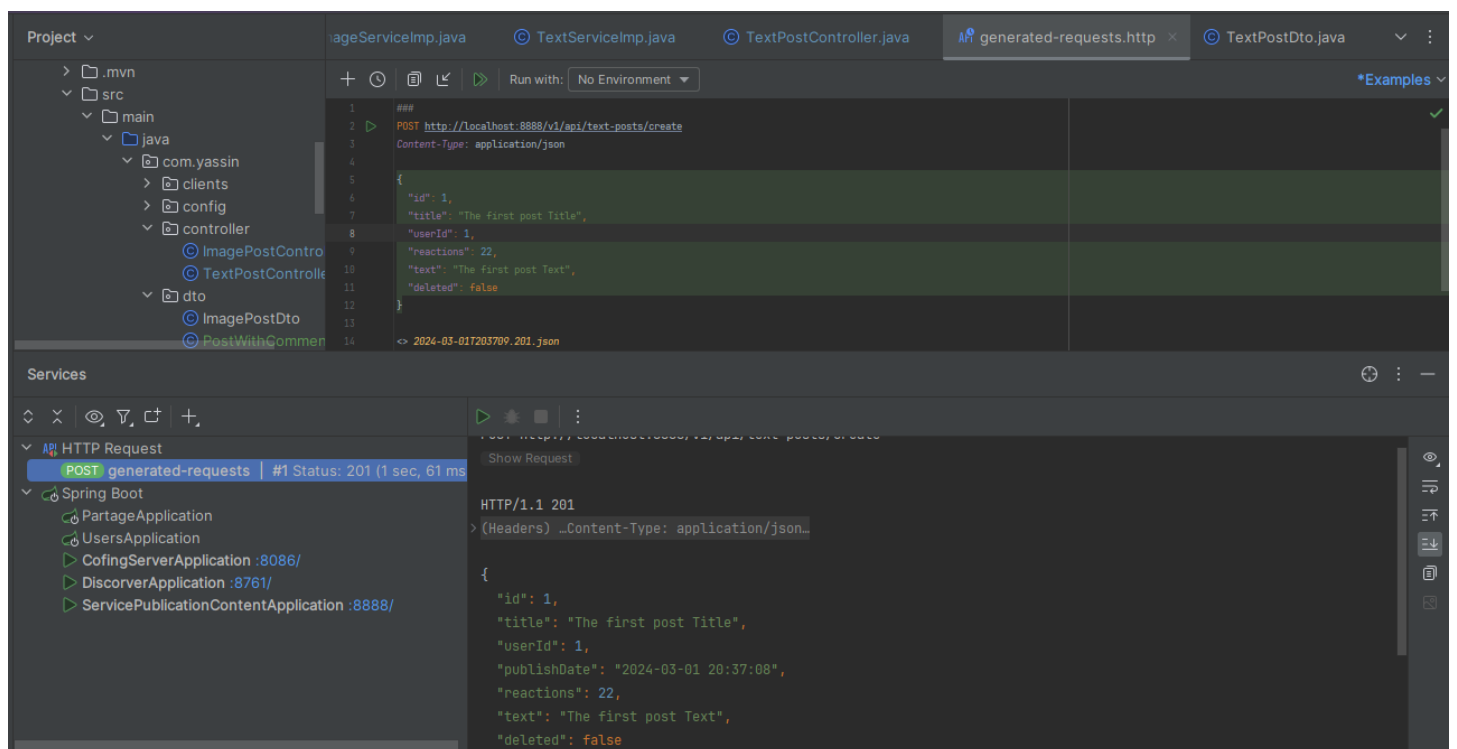
Post Service

Dans notre application, le service de publication joue un rôle crucial pour permettre aux utilisateurs d'interagir et de partager du contenu avec leur réseau. Les fonctionnalités de base de ce service incluent la possibilité de créer des publications textuelles ou des publications contenant des images. Les utilisateurs peuvent rédiger des commentaires personnalisés.

En plus de la création de publications, les utilisateurs peuvent généralement interagir avec les publications des autres en les aimant, en les commentant ou en les partageant. Ces interactions favorisent l'engagement et la communication au sein de la communauté virtuelle. Certains services de médias sociaux offrent également des fonctionnalités avancées telles que la programmation de publications, la création de sondages, ou même la possibilité de diffuser en direct des vidéos pour une expérience plus interactive.

Voice quelque fonctionnalités basic de ce Service

1. Ajouter un Post:



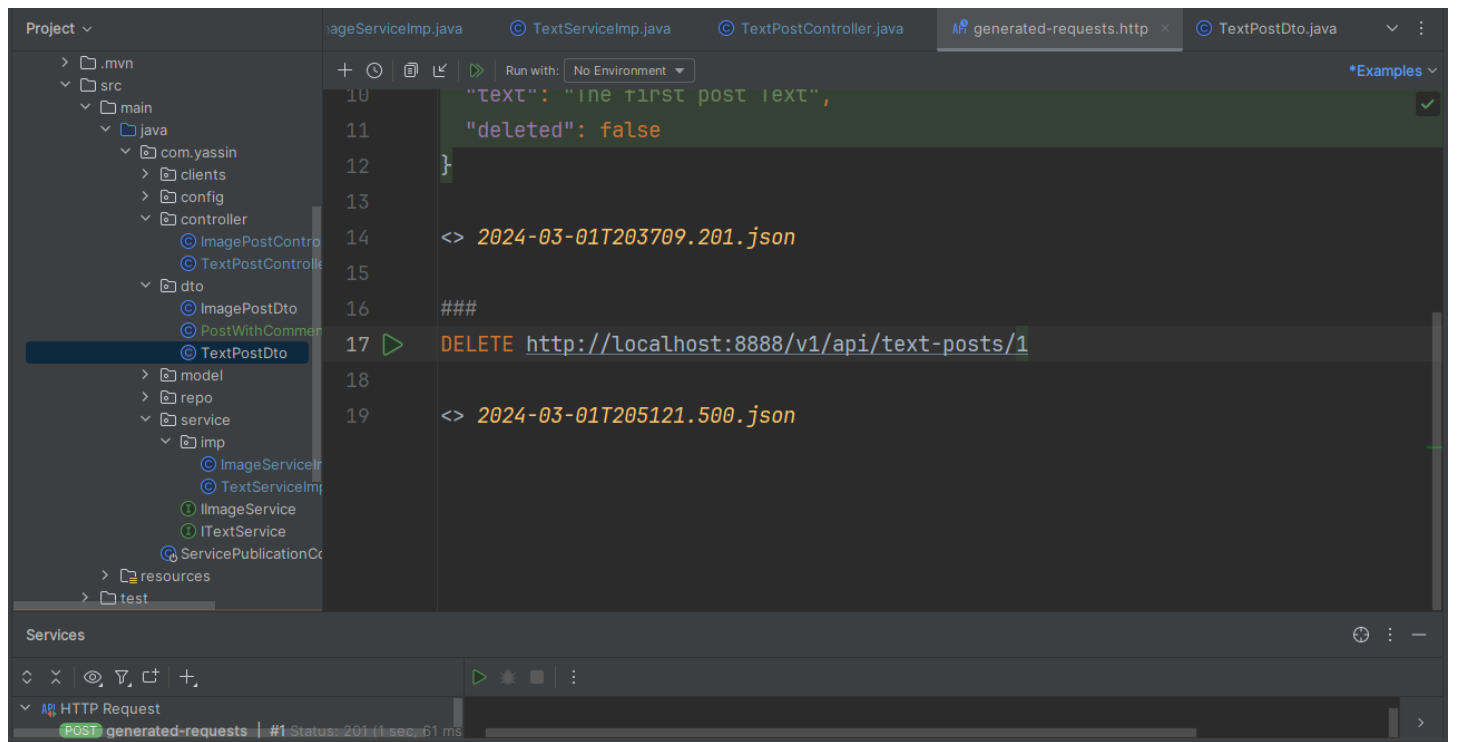
The screenshot displays an IDE interface with a REST client tab titled 'generated-requests.http'. The request is a POST to 'http://localhost:8888/v1/api/text-posts/create' with a JSON body. The response is a 201 status code with a JSON body containing post details.

```
###
1 POST http://localhost:8888/v1/api/text-posts/create
2 Content-Type: application/json
3
4 {
5   "id": 1,
6   "title": "The first post Title",
7   "userId": 1,
8   "reactions": 22,
9   "text": "The first post Text",
10  "deleted": false
11 }
12
13
14 < 2024-03-01T20:37:09.201.json
```

The Services panel on the left shows the 'HTTP Request' tab with the following details:

- Method: POST
- URL: http://localhost:8888/v1/api/text-posts/create
- Status: 201 (1 sec, 61 ms)
- Headers: Content-Type: application/json
- Body: { "id": 1, "title": "The first post Title", "userId": 1, "publishDate": "2024-03-01 20:37:08", "reactions": 22, "text": "The first post Text", "deleted": false }

1. supprimer un post:



1. update un post.

