

# PLATEFORME DE GESTION D'UNE ÉCOLE DE LANGUE ANGLAISE



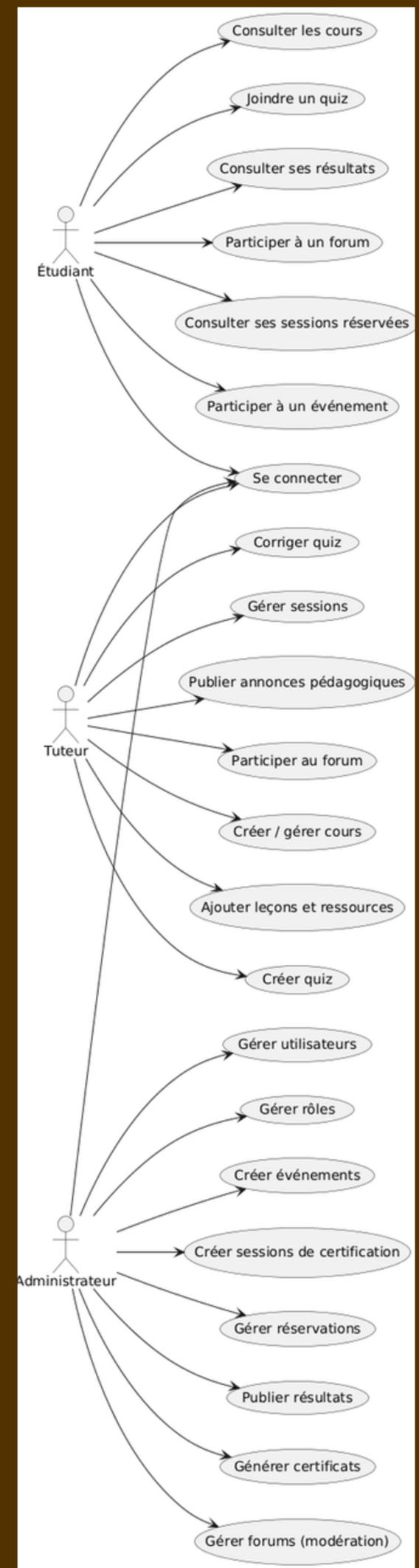
# Objectif du projet:



Le projet consiste à concevoir et développer une plateforme web de gestion d'une école de langue anglaise, destinée aux étudiants.

La plateforme permet la gestion des cours, événements, forums, quiz, ainsi que des sessions et réservations, en s'appuyant sur une architecture microservices afin d'assurer la scalabilité, la maintenabilité et la séparation des responsabilités.

# diagramme de cas d'utilisation



## **4. Description des microservices:**

## 4. Description des microservices (complet)

Nom du microservice	Responsabilité	Fonctionnalités principales	Technologie utilisée
 <a href="#"><u>User Service</u></a>	Gestion des utilisateurs et sécurité	<ul style="list-style-type: none"><li>• Inscription / connexion</li><li>• Gestion des profils</li><li>• Gestion des rôles</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• Spring Security / JWT</li><li>• MySQL / PostgreSQL</li></ul>
 <a href="#"><u>Course Service</u></a>	Gestion des cours	<ul style="list-style-type: none"><li>• Création et gestion des cours</li><li>• Modules et leçons</li><li>• Affectation aux utilisateurs</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• Spring Security / JWT</li><li>• MySQL / PostgreSQL</li></ul>
 <a href="#"><u>Event Service</u></a>	Gestion des événements	<ul style="list-style-type: none"><li>• Planification des événements</li><li>• Gestion des participants</li><li>• Calendrier</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• REST API</li><li>• MySQL / PostgreSQL</li></ul>
 <a href="#"><u>Forum Service</u></a>	Gestion des discussions	<ul style="list-style-type: none"><li>• Sujets et messages</li><li>• Commentaires</li><li>• Modération</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• REST API</li><li>• Base relationnelle</li></ul>
 <a href="#"><u>Quiz Service</u></a>	Évaluation des utilisateurs	<ul style="list-style-type: none"><li>• Création de quiz</li><li>• Soumission des réponses</li><li>• Calcul des scores</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• REST API</li><li>• Message Broker (optionnel)</li></ul>
 <a href="#"><u>Session &amp; Reservation Service</u></a>	Gestion des sessions et réservations	<ul style="list-style-type: none"><li>• Création des sessions</li><li>• Réservations / annulations</li><li>• Gestion des disponibilités</li></ul>	<ul style="list-style-type: none"><li>• Spring Boot</li><li>• REST API</li><li>• Base relationnelle</li></ul>

# 5. Communication & flux:

## 5&gt;

## Communication & Flux



### Types de communication

#### > ✓ Communication synchrone (REST)

- Protocole : HTTP / HTTPS
- Format : JSON
- Utilisée pour les appels directs via l'API Gateway

**Exemple :**

- Session & Reservation Service → **User Service**

#### > + Communication asynchrone (optionnelle)

- Via un **Message Broker** (Kafka ou RabbitMQ)
- Pour le traitement d'événements

**Exemple :**

- Quiz Service → Notifications / **Certifications**

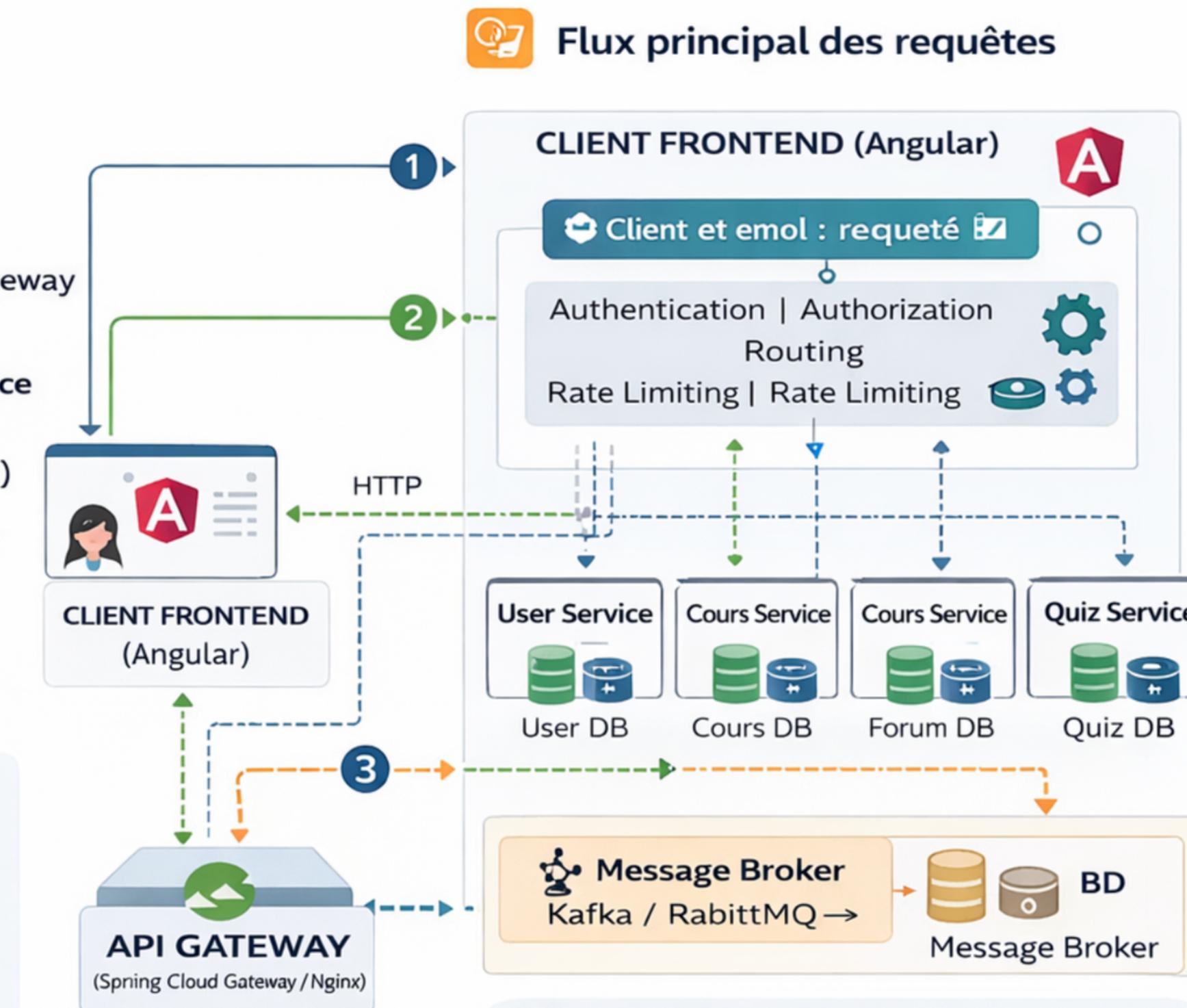


### Flux principal des requêtes

1. L'utilisateur envoie une requête
2. Autorisation par rôles (Étudiant, Tuteur, ...)
3. Comm. sécurisée via HTTPS



### Flux principal des requêtes



### Sécurité des flux

- Authentification basée sur JWT
- Autorisation par rôles (Étudiant, Tuteur, ...)
- Comm. sécurisée via HTTPS

# Infrastructure & déploiement:

## 🐳 Conteneurisation

- Chaque microservice est déployé dans un conteneur Docker
- Déploiement indépendant
- Isolation des pannes





# Bases de données

Une base de données par microservice :

- **UserDB**
- **CourseDB**
- **EventDB**
- **ForumDB**
- **QuizDB**
- **SessionDB**



## Communication avec le Frontend :

- Le frontend communique uniquement avec l'API Gateway
- Les microservices ne sont jamais exposés directement

## Conclusion:

Notre architecture microservices découpe l'application en services autonomes, communiquant via REST et un broker asynchrone. L'API Gateway sécurise l'accès avec JWT. Cette structure garantit flexibilité, évolutivité et maintenance simplifiée.



# THANK YOU

---

