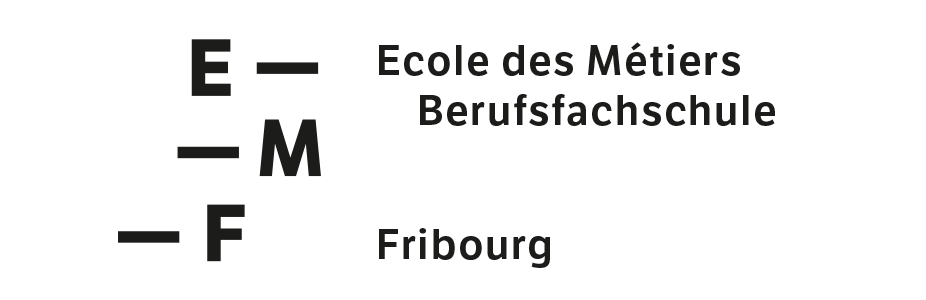
Module 133

Réaliser des applications Web en Sessions-Handling

Illan Angel & Noam Bourqui



Documentation de projet

Module du 17.03.2025 – xx.xx.xx

Table des matières

[Analyse à faire complètement avec EA 4](#_Toc194309126)

[Use case client et use case Rest 4](#_Toc194309127)

[Use case client 4](#_Toc194309128)

[Use case Admin 5](#_Toc194309129)

[Use case APIGateway 6](#_Toc194309130)

[Use case rest Client 7](#_Toc194309131)

[Use case rest Admin 8](#_Toc194309132)

[Activity Diagram d'un cas complet navigant dans les applications avec les explications 9](#_Toc194309133)

[Activity Diagram Register 9](#_Toc194309134)

[Maquettes du projet avec la navigation intégrée et ses explications 10](#_Toc194309135)

[Maquette Client 10](#_Toc194309136)

[Maquette Admin 11](#_Toc194309137)

[Sequence System global entre les applications 12](#_Toc194309138)

[Register 12](#_Toc194309139)

[Login 13](#_Toc194309140)

[Conception à faire complétement avec EA 15](#_Toc194309141)

[Class Diagram complet avec les explications de chaque application 15](#_Toc194309142)

[Navigation Diagram complète avec les explications des applications 18](#_Toc194309143)

[Navigation APIREST 1 18](#_Toc194309144)

[Navigation APIREST 2 19](#_Toc194309145)

[Bases de données 19](#_Toc194309146)

[Modèles ER 20](#_Toc194309147)

[Modèles ER client 20](#_Toc194309148)

[Modèles ER Admin 20](#_Toc194309149)

[Modèles WorkBench MySQL 21](#_Toc194309150)

[Modèles Workbench MYSQL Client 21](#_Toc194309151)

[Table Categorie 21](#_Toc194309152)

[Table Question 22](#_Toc194309153)

[Table Utilisateur 22](#_Toc194309154)

[Modèles Workbench MYSQL Admin 23](#_Toc194309155)

[Table Administrateur 23](#_Toc194309156)

[Implémentation des applications 23](#_Toc194309157)

[Une descente de code client 23](#_Toc194309158)

[Implémentation de l'application 23](#_Toc194309159)

[Une descente de code APIGateway 23](#_Toc194309160)

[Implémentation des applications <API élève1> et <API élève2> 23](#_Toc194309161)

[Une descente de code de l'API REST 23](#_Toc194309162)

[Hébergement 23](#_Toc194309163)

[Installation du projet complet avec les 5 applications 24](#_Toc194309164)

[Outils, langages (versions, définitions, installations) 24](#_Toc194309165)

[Tests de fonctionnement du projet 24](#_Toc194309166)

[Auto-évaluations et conclusions 24](#_Toc194309167)

[Auto-évaluation et conclusion de Illan Angel 24](#_Toc194309168)

[Auto-évaluation et conclusion de Noam Bourqui 24](#_Toc194309169)

Introduction et contexte du project

Le projet consiste en la création d'un quiz informatique interactif couvrant des thématiques telles que le développement, les réseaux, les systèmes et la sécurité. L'utilisateur pourra choisir un thème parmi ceux proposés, puis lancer une partie dans laquelle il sera confronté à une série de questions auxquelles il devra répondre. Selon ses réponses, un score total sera calculé pour refléter ses connaissances. De plus, un classement des meilleurs utilisateurs sera mis en place, permettant de voir les participants les plus performants.

Une fonctionnalité essentielle du projet sera la partie administrateur. En tant qu'administrateurs, les utilisateurs auront la possibilité d'ajouter de nouveaux quiz, de créer de nouveaux thèmes – par exemple, un thème sur l'intelligence artificielle – et d'ajouter de nouvelles questions pour enrichir l'expérience des utilisateurs. Ce projet vise à offrir une expérience ludique tout en favorisant l'apprentissage des concepts clés dans différents domaines de l'informatique.

# Analyse à faire complètement avec EA

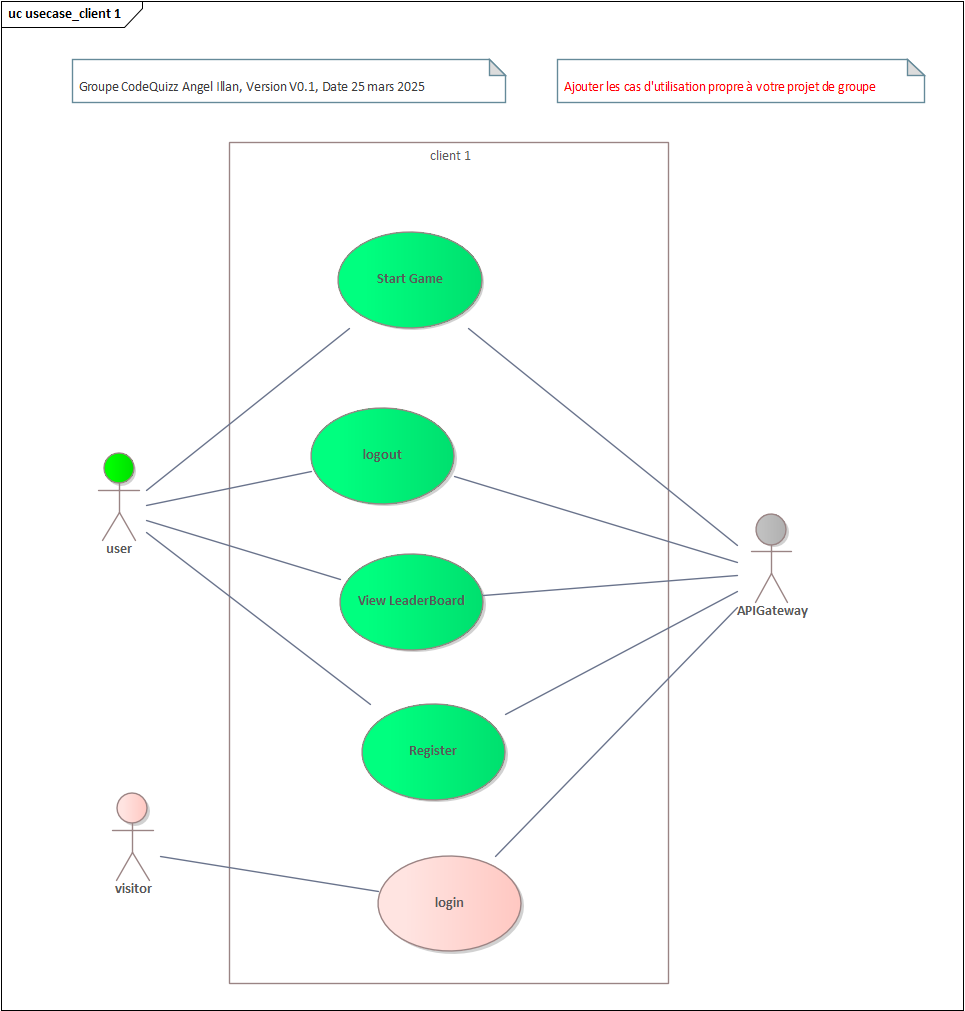
Dans cette section, nous allons nous concentrer sur l’analyse approfondie du projet. Nous explorerons les différents diagrammes qui permettront de mieux comprendre son fonctionnement et son évolution. Cela inclut des schémas de type Use Case, des diagrammes d’activités, des diagrammes de séquence, ainsi que des maquettes visuelles.

Ces outils seront essentiels pour illustrer les interactions, les flux et la structure du projet, afin de nous donner une vision claire et concrète de ce à quoi il ressemblera une fois terminé. L’objectif est de permettre à tous les intervenants de se projeter dans le résultat final et de mieux saisir les différentes étapes de son développement.

## Use case client et use case Rest

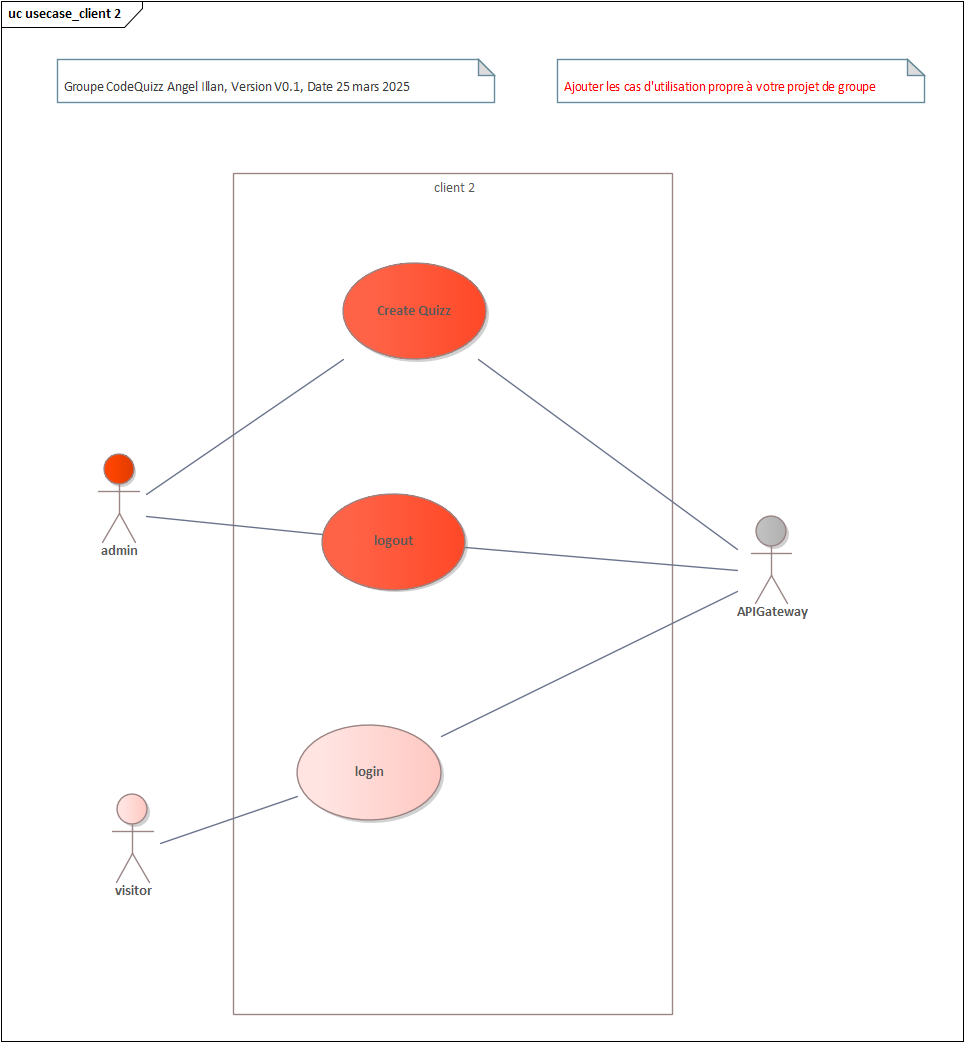
### Use case client

Dans le diagramme ci-dessous, nous présentons un diagramme de classe du côté client, représentant l'utilisateur normal. Ce dernier dispose de plusieurs fonctionnalités accessibles : il peut se connecter ou s'enregistrer, démarrer une nouvelle partie, ou encore consulter le classement des autres utilisateurs. Ce diagramme permet de visualiser les interactions et les différentes actions que l'utilisateur peut entreprendre au sein de l'application, tout en mettant en évidence les relations entre les différentes classes impliquées dans ces processus. Il offre ainsi une vue d'ensemble du fonctionnement et des options disponibles pour l'utilisateur final.



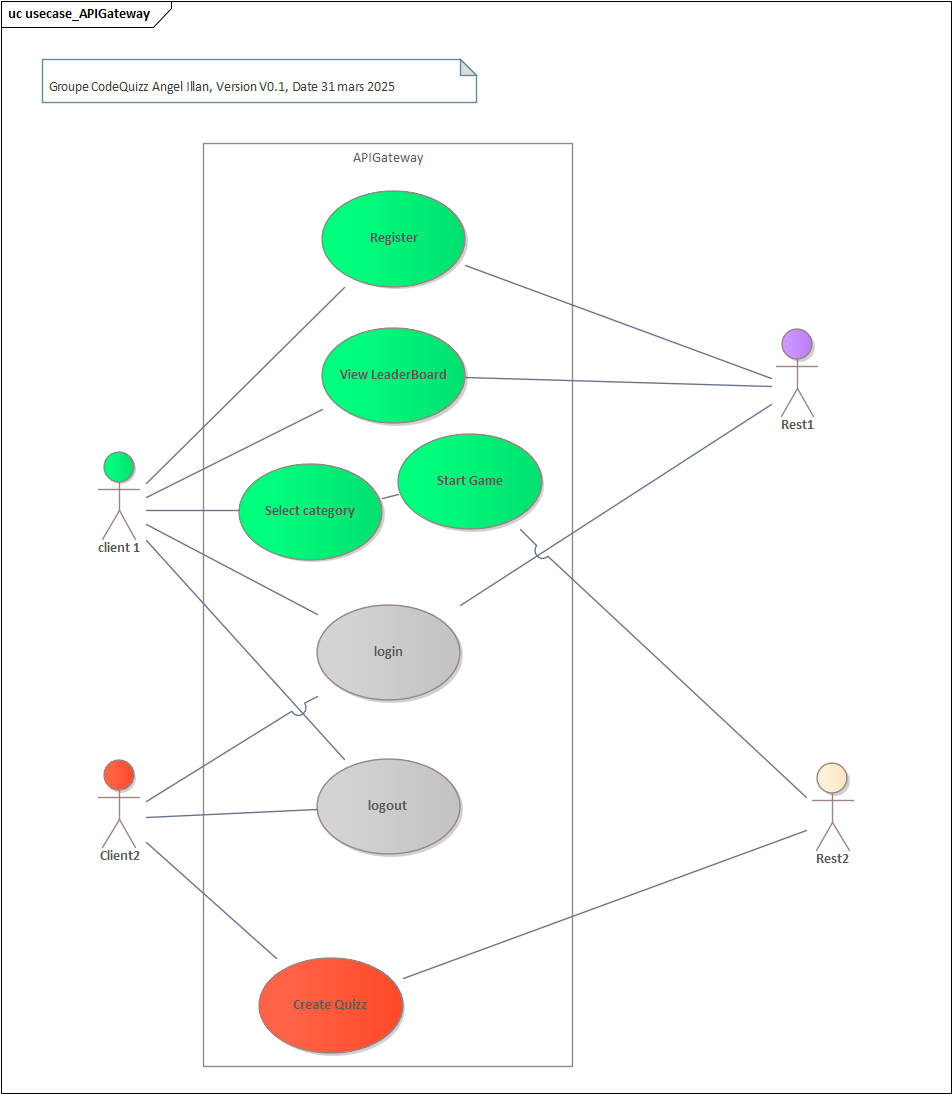
### Use case Admin

Dans le diagramme ci-dessous, nous illustrons la partie dédiée à l'administrateur. Comme l'utilisateur normal, l'administrateur peut également se connecter à la plateforme. Cependant, ses fonctionnalités vont bien au-delà, puisqu'il a la possibilité de créer de nouveaux quiz. Ce diagramme met en évidence les actions spécifiques que l'administrateur peut réaliser, ainsi que les interactions entre les différentes entités qui lui permettent de gérer et de configurer le contenu de l'application.



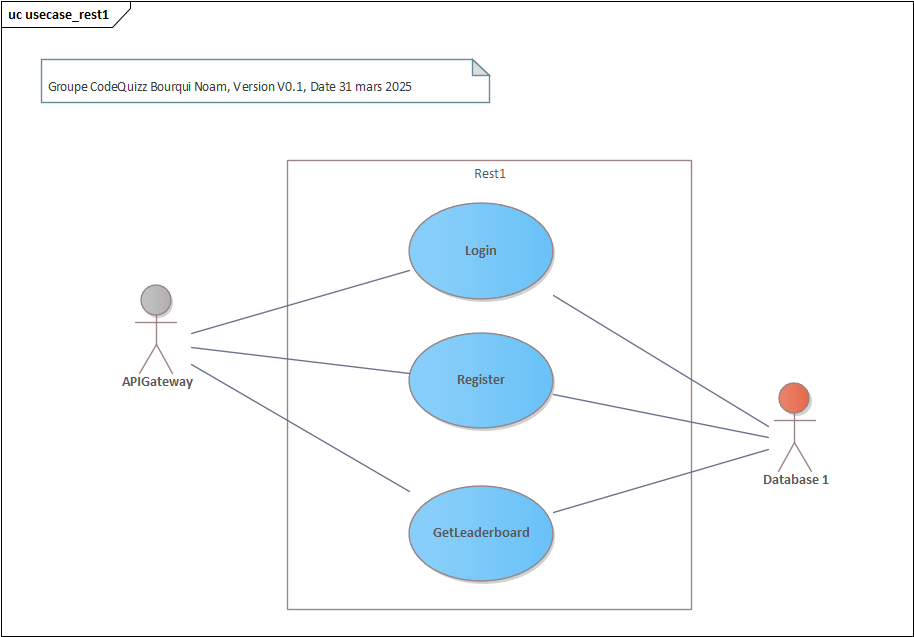
### Use case APIGateway

Dans ce schéma, nous voyons l'ensemble des actions que peuvent réaliser les utilisateurs et les administrateurs, qui interagissent avec la partie backend du projet via une architecture REST. Ce diagramme représente l'API Gateway, qui joue un rôle central en dirigeant les différentes requêtes des utilisateurs et des administrateurs vers les services appropriés du backend. Il permet de centraliser les échanges entre le frontend et le backend, facilitant ainsi la gestion des appels API et le traitement des données.



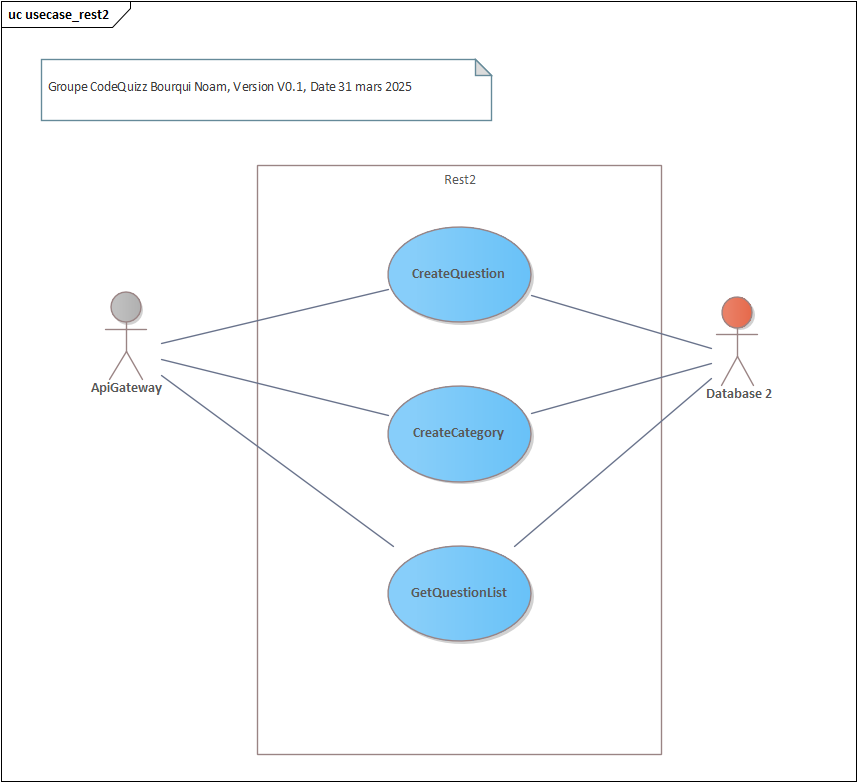
### Use case rest Client

Dans cette section, nous examinons ce qui se passe après l'API Gateway du côté des utilisateurs normaux. Tout d'abord, l'utilisateur/admin peut se connecter à son compte ou s'enregistrer s'il n'en possède pas encore. Une fois connecté et après avoir lancé une partie, il accède aux questions du quiz, qui lui sont présentées une à une. En parallèle, il peut consulter le classement des joueurs dans le leaderboard, afin de suivre sa position par rapport aux autres participants. Ce processus montre l'interaction de l'utilisateur avec le système, permettant de gérer ses actions tout en maintenant une expérience fluide et continue, allant de la connexion à la participation active au quiz et à la consultation des résultats.



### Use case rest Admin

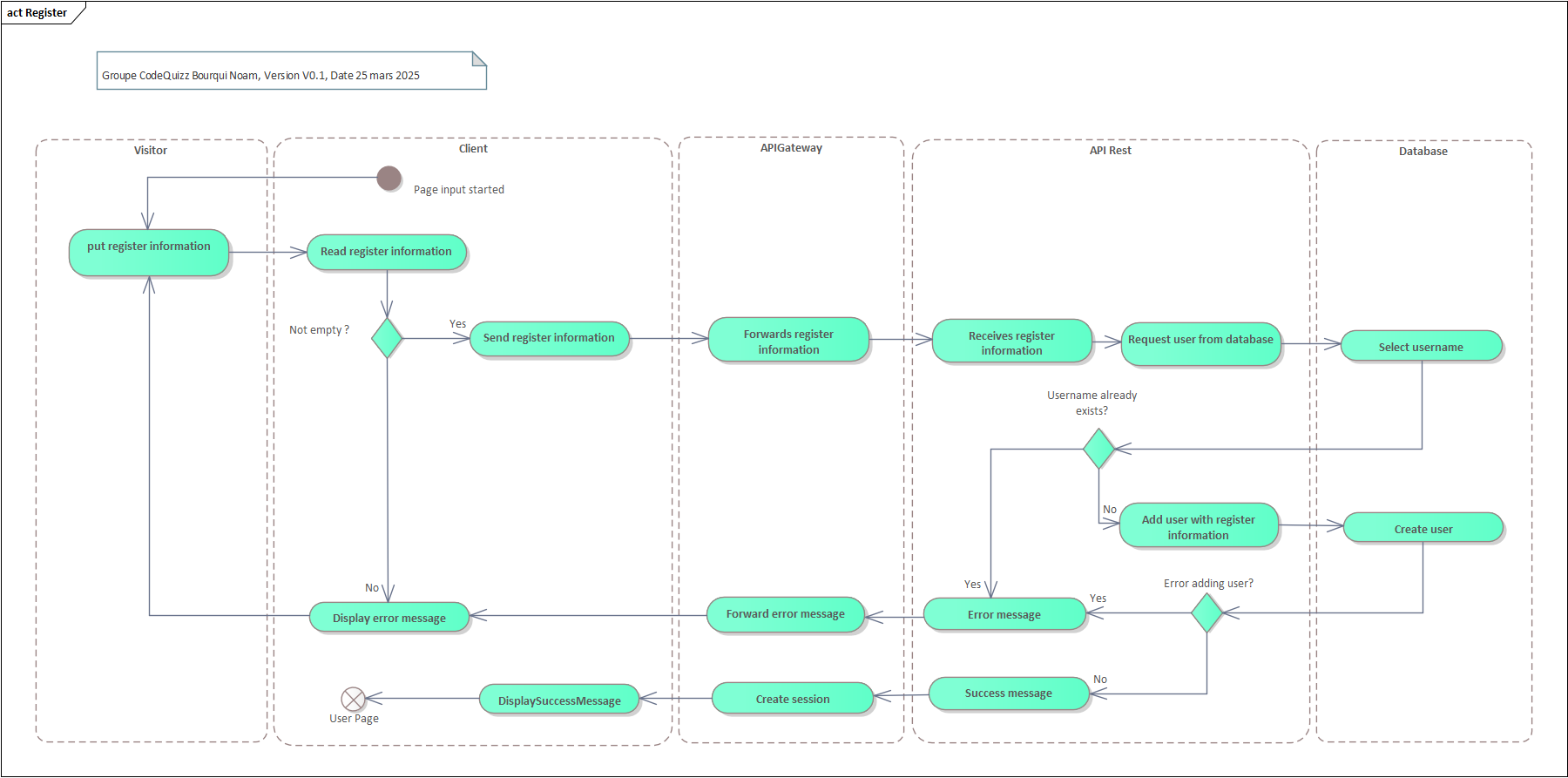
L'API Gateway permettra à l'administrateur de créer des catégories avec de nouvelles question, en définissant ses choix de réponses correspondants, puis de l'ajouter directement dans la base de données. Ensuite, l’api rest devra créer la liste de question pour le frontend. Cette partie du processus illustre comment l'API Gateway facilite la gestion du contenu par l'administrateur, en assurant l'enregistrement des nouvelles questions dans le système et en offrant un accès sécurisé aux options de gestion.



## Activity Diagram d'un cas complet navigant dans les applications avec les explications

### Activity Diagram Register

Ce diagramme illustre les étapes du processus d'inscription d'un utilisateur. L'utilisateur entre ses informations d'inscription, qui sont ensuite vérifiées par le client. Si elles sont valides, elles sont envoyées à l'API Gateway, puis transmises à l'API Rest. L’API vérifie si le nom d'utilisateur est déjà utilisé en consultant la base de données. Si le nom est libre, le compte est créé et une session est ouverte. Sinon, un message d'erreur est retourné. D'autres erreurs peuvent être générées si des champs sont vides ou si un problème survient lors de l'ajout en base de données.



## Maquettes du projet avec la navigation intégrée et ses explications

### Maquette Client

Premièrement, l’utilisateur devra se login avec un compte déjà existant ou se créer un compte. Ensuite, il aura le choix des catégories pour les questions du quizz. Une fois qu’il aura cliqué sur une des catégories la partie se lancera automatiquement. Il a également la possibilité d’afficher le classement des points entre tous les utilisateurs. Quand il sera en partie, il aura 4 choix de réponse possible par questions. Et pour finir, il pourra se logout.



### Maquette Admin

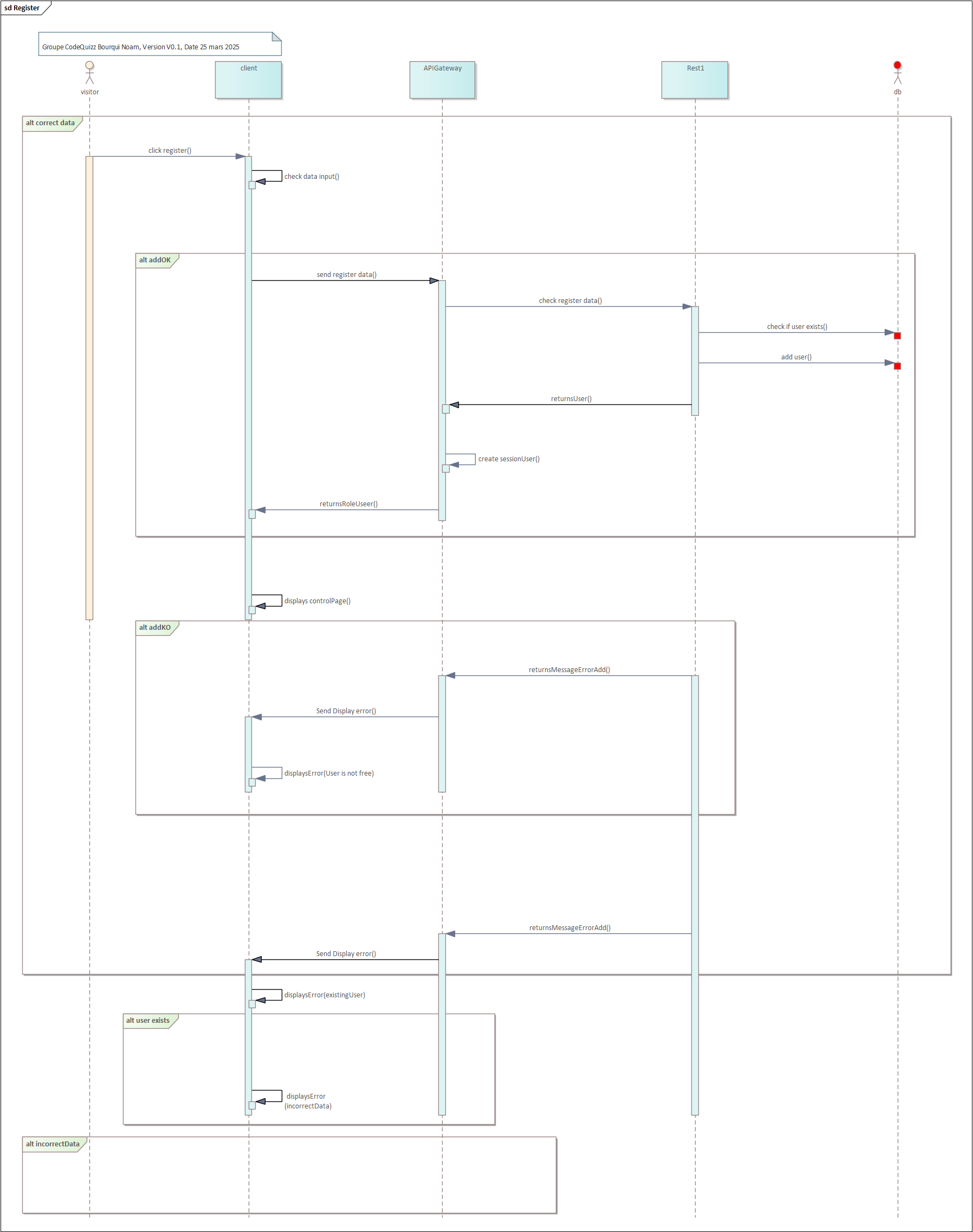
Le visiteur devra se connecter avec un compte administrateur. Une fois connecté, il pourra créer des questions en sélectionnant la catégorie, en saisissant la question et en définissant les choix de réponses. Il aura également la possibilité de créer de nouvelles catégories de questions. Enfin, il pourra se déconnecter à tout moment.



## Sequence System global entre les applications

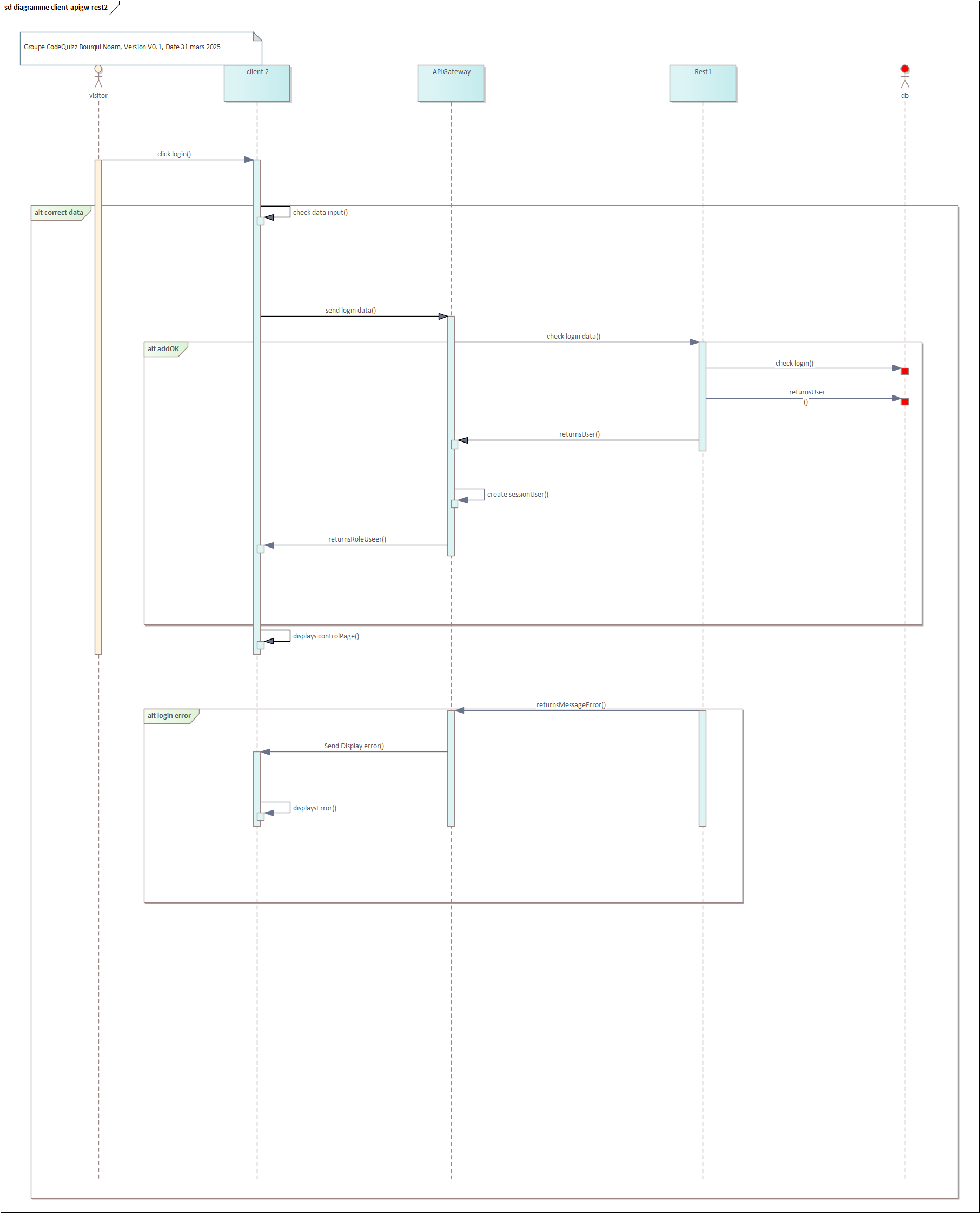
### Register

Ce diagramme détaille les échanges entre les différentes entités du système lors de l'inscription. L'utilisateur clique sur "Register", ce qui déclenche une vérification des données par le client. Si elles sont correctes, elles sont envoyées à l'API Gateway, qui les transmet à l'API Rest. L'API vérifie si l'utilisateur existe déjà et, si ce n'est pas le cas, ajoute le nouvel utilisateur en base de données avant de créer une session et de renvoyer une confirmation. En cas d'erreur (nom déjà utilisé, données invalides ou problème technique), un message d'erreur est retourné au client, qui l'affiche à l'utilisateur.



### Login

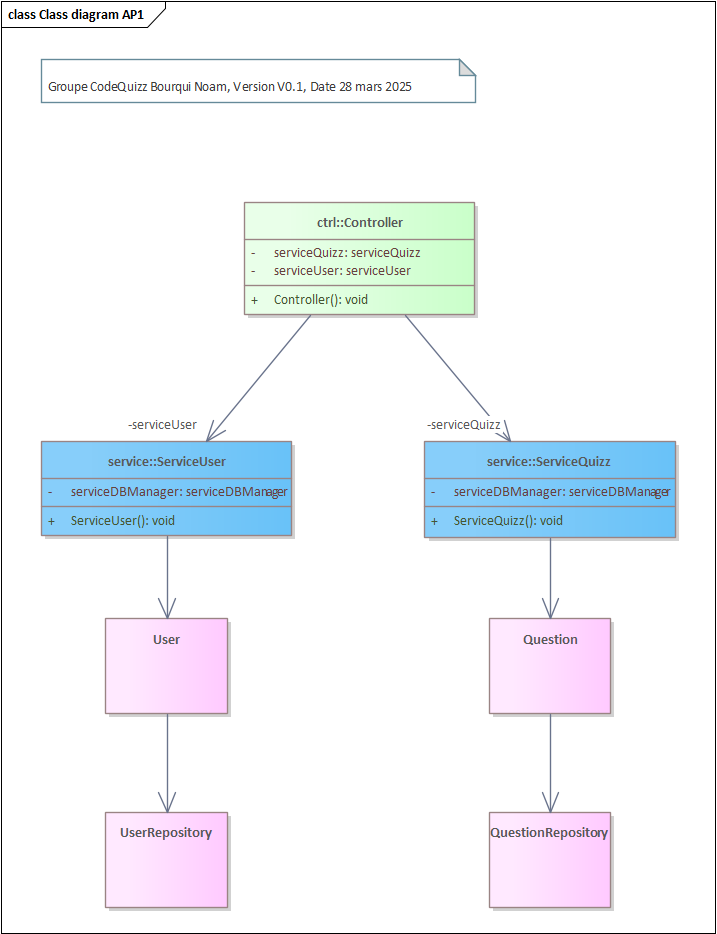
Ce diagramme décrit les échanges entre les différentes entités du système lors de la tentative de connexion. L'utilisateur clique sur "Login", ce qui déclenche une vérification des données par le client. Si les informations sont valides, elles sont envoyées à l'API Gateway, qui les transmet à l'API Rest. L'API vérifie si l'utilisateur existe et si le mot de passe correspond. Si les informations sont correctes, une session est créée et l'utilisateur est connecté. En cas d'erreur (identifiants incorrects, problème technique, etc.), un message d'erreur est renvoyé au client, qui l'affiche à l'utilisateur.



# Conception à faire complétement avec EA

## Class Diagram complet avec les explications de chaque application

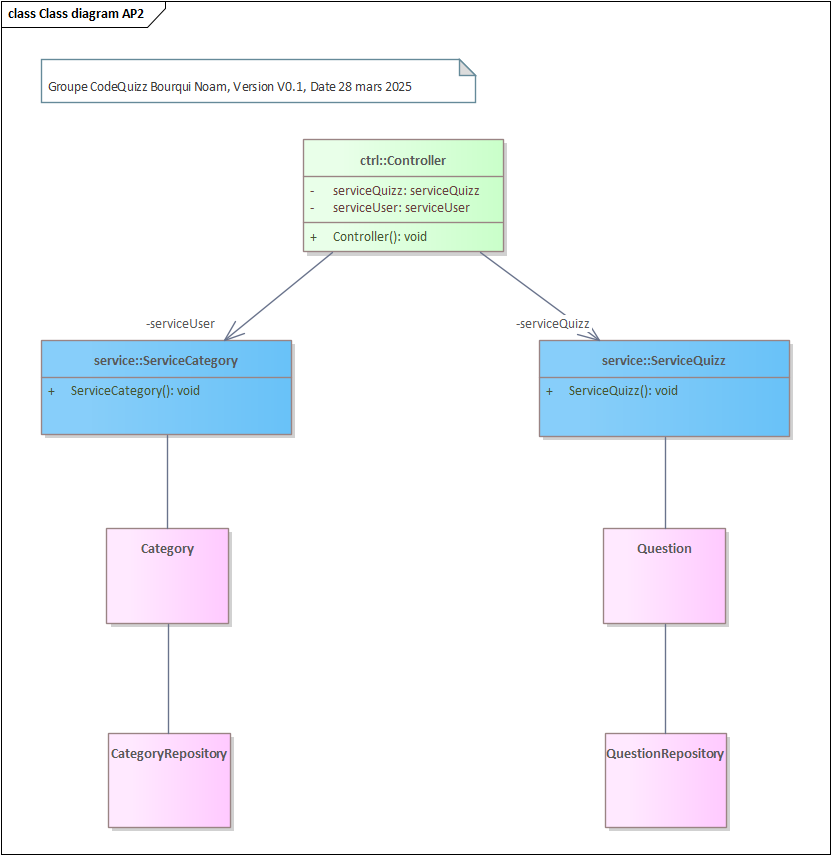
Ce schéma de classe montre comment fonctionne CodeQuizz, une application qui gère les utilisateurs et les quiz. Au centre, on a un Controller qui fait le lien entre deux services : ServiceUser pour les utilisateurs et ServiceQuizz pour les quiz. Chaque service communique avec une base de données (serviceDBManager) et gère ses propres entités. ServiceUser s’occupe des utilisateurs, stockés dans UserRepository, tandis que ServiceQuizz gère les questions, enregistrées dans QuestionRepository (Spring-boot). L’architecture est bien organisée : le contrôleur gère les interactions, les services appliquent la logique métier, et les repositories stockent les données. Ce modèle rend l’application facile à comprendre et à faire évoluer.



Ce schéma de classe représente une version administrateur de l’application CodeQuizz, mais avec une différence importante : l’administrateur peut créer et gérer des catégories et des questions, mais il ne gère pas les utilisateurs.

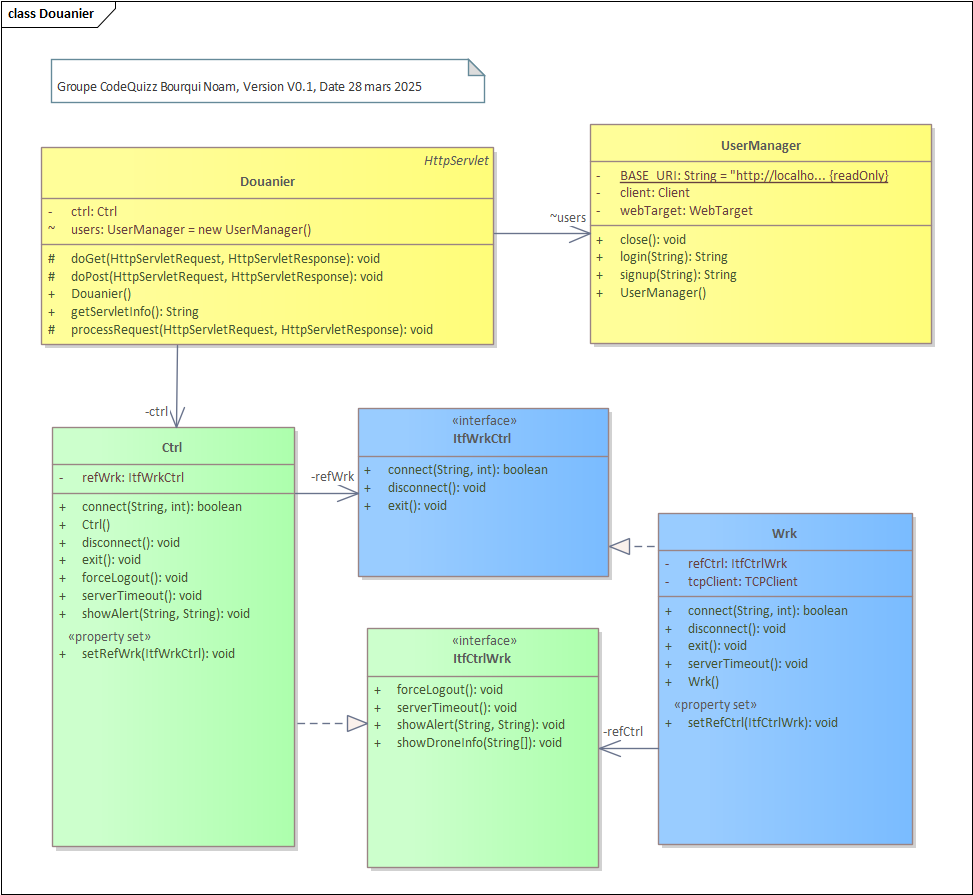
Au centre, le Controller coordonne deux services : ServiceCategory pour gérer les catégories et ServiceQuizz pour gérer les questions. ServiceCategory permet d'ajouter, modifier ou supprimer des catégories, qui sont stockées dans CategoryRepository. De son côté, ServiceQuizz gère les questions, enregistrées dans QuestionRepository.

Cette architecture est bien structurée, avec une séparation claire des responsabilités. L’administrateur a uniquement accès à la gestion du contenu (questions et catégories), ce qui empêche toute modification des utilisateurs et garantit une meilleure organisation du système.



Ce schéma représente l’API Gateway de l’application CodeQuizz. Son rôle est de rediriger les requêtes des utilisateurs vers la bonne API en fonction de leur rôle, des actions demandées et de gérer la session des utilisateurs.

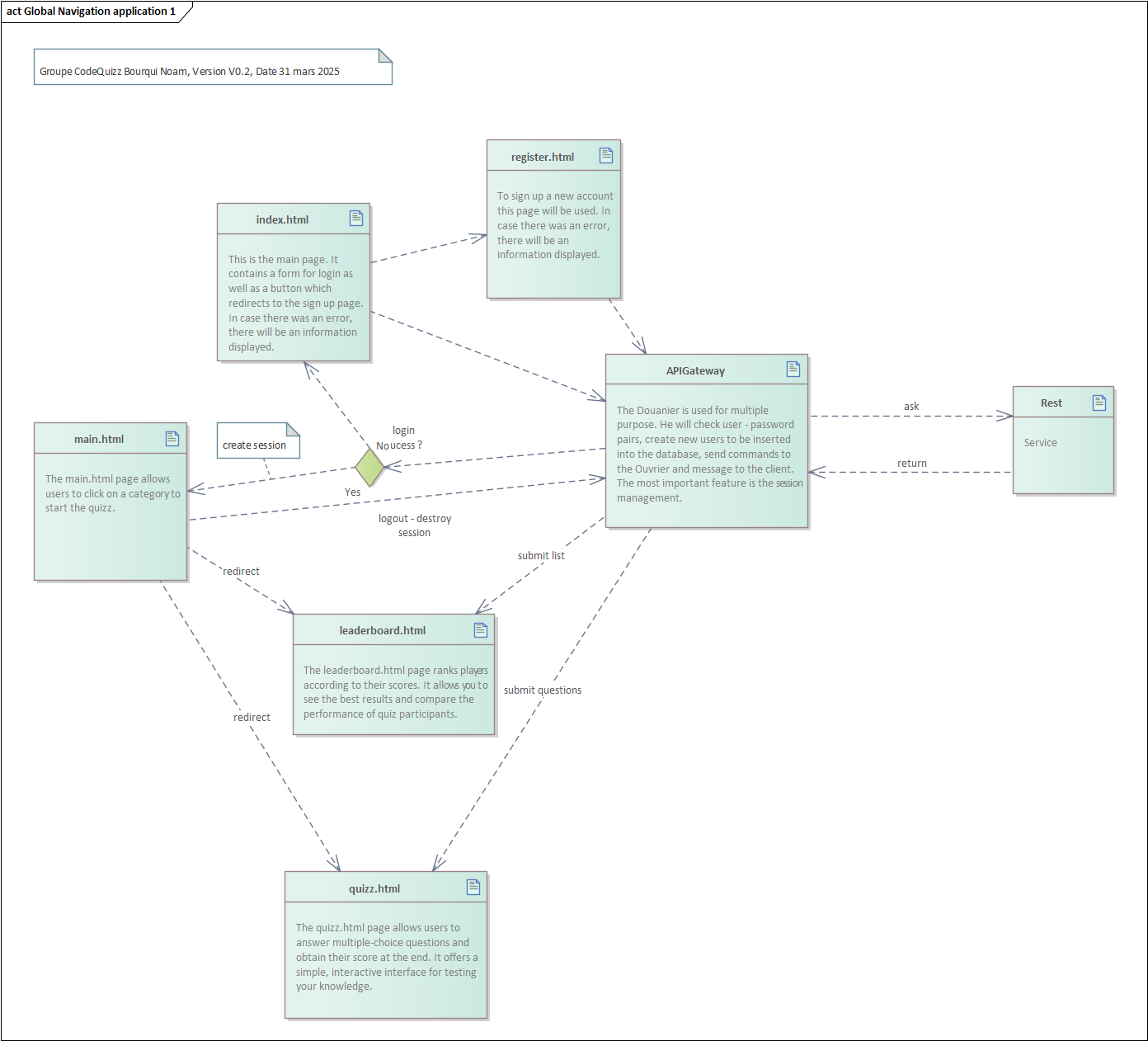
Cette architecture permet une meilleure sécurité et organisation en filtrant les accès dès l'entrée dans le système. L'API Gateway agit donc comme un point unique d’entrée et garantit que chaque requête est dirigée vers l’API appropriée en fonction des droits de l’utilisateur.



## Navigation Diagram complète avec les explications des applications

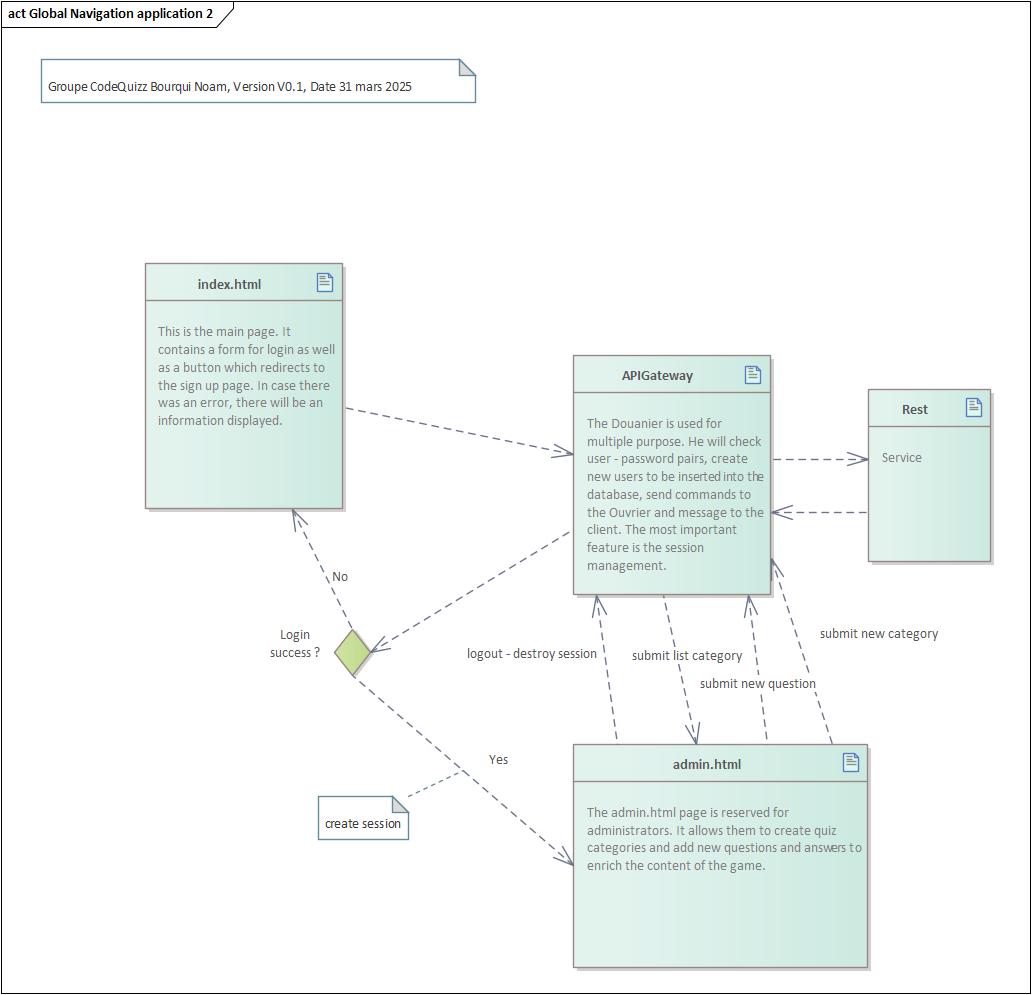
### Navigation APIREST 1

Sur ce schéma, l'utilisateur peut soit se connecter s'il possède déjà un compte, soit s'enregistrer s'il n'en a pas encore. Une fois connecté, il pourra choisir une catégorie de quiz parmi celles proposées. Après avoir sélectionné une catégorie, il pourra lancer le quiz. À la fin de celui-ci, un score sera calculé en fonction de ses réponses et de sa performance. L'utilisateur aura également la possibilité de consulter le leaderboard pour voir les meilleurs joueurs. Ce diagramme démontre également la gestion des sessions.



### Navigation APIREST 2

Sur ce schéma, le visiteur devra se connecter, si le login est correct il pourra accéder à l’interface administrateur (admin.html). Il pourra créer des catégories ou créer des questions. Nous voyons également que le admin.html interagis avec l’api gateway pour tout ce qui est backend. Ce diagramme comprend également le session-handling.



# Bases de données

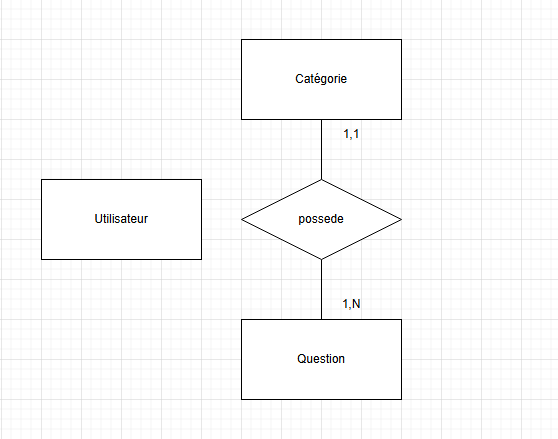
La base de données est conçue pour gérer un quiz interactif avec des questions classées par catégories. Elle permet aux utilisateurs de répondre aux questions et aux administrateurs de gérer le contenu.

## Modèles ER

Les modèles Entité-Relation (ER) définissent la structure des relations entre les différentes entités de la base de données.

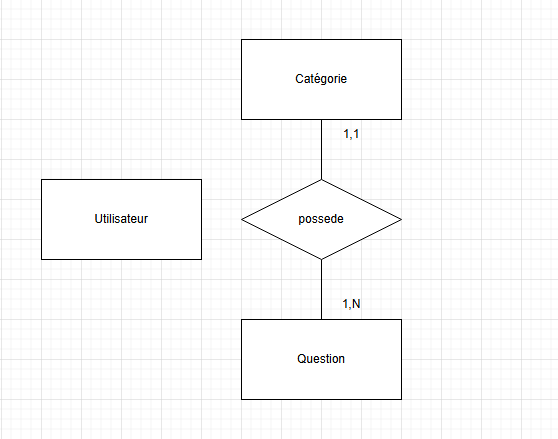
### Modèles ER client

Ce modèle concerne les utilisateurs du site.



### Modèles ER Admin

Ce modèle concerne les questions avec les différentes catégories auxquelles elles appartiennent.

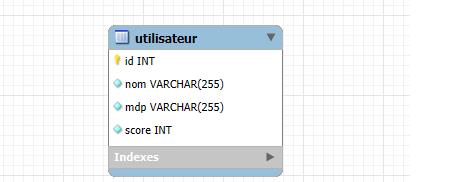


## Modèles WorkBench MySQL

### Modèles Workbench MYSQL Client

#### Table Administrateur

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Contraintes | Description |
| Id | INT | PRIMARY KEY, AUTO\_INCREMENT | Identifiant unique de l'administrateur |
| nom | |  |  | | --- | --- | | VARCHAR(255) |  | | NOT NULL | Nom de l’administrateur |
| mdp | |  |  | | --- | --- | | VARCHAR(255) |  | | NOT NULL | Mot de passe de l’administrateur |



### Modèles Workbench MYSQL Admin

#### Table Categorie

Stocke les différentes catégories de questions du quiz.

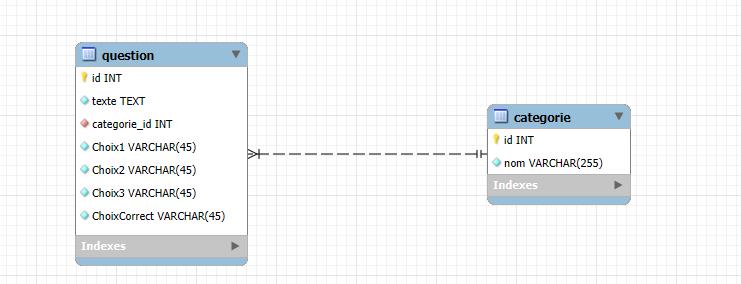
|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Contraintes | Description |
| Id | INT | PRIMARY KEY, AUTO\_INCREMENT | Identifiant unique de la catégorie |
| Nom | VARCHAR(255) | NOT NULL | Nom de la catégorie |

#### Table Question

Stocke les questions associées aux catégories.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Contraintes | Description |
| Id | INT | PRIMARY KEY, AUTO\_INCREMENT | Identifiant unique de la question |
| Texte | TEXT | NOT NULL | Contenue de la question |
| Categorie\_id | INT | NOT NULL, FOREIGN KEY | |  |  | | --- | --- | |  | Identifiant de la catégorie associée | |
| choix\_1 | TEXT | NOT NULL | Identifiant de la catégorie associée |
| choix\_2 | TEXT | NOT NULL | Deuxième choix possible |
| choix\_3 | TEXT | NOT NULL | Troisième choix possible |
| Choix\_4 | TEXT | NOT NULL | Quatrième choix possible |
| Bonne\_reponse | INT | NOT NULL CHECK (bonne\_reponse IN (1,2,3,4)) | Numéro du choix correct |

* Une catégorie peut posséder plusieurs questions (1,N).
* Une question appartient à une seule catégorie (1,1).



# Implémentation des applications

## Une descente de code client

# Implémentation de l'application

## Une descente de code APIGateway

# Implémentation des applications

## Une descente de code de l'API REST

# Hébergement

# Installation du projet complet avec les 5 applications

# Outils, langages (versions, définitions, installations)

# Tests de fonctionnement du projet

# Auto-évaluations et conclusions

## Auto-évaluation et conclusion de Illan Angel

## Auto-évaluation et conclusion de Noam Bourqui