| ***CASTANO ORTIZ Diana Carolina*** |
| --- |
| ***Dossier Projet*** |
| ***CDA 2022- 2023 / MNS / IFA*** |
| ***CASTANO ORTIZ Diana Carolina***  ***31/07/2023*** |

***Table des matières***

***[1. Liste des compétences du référentiel qui sont couvertes par le projet](#_gjdgxs) 2***

***[2. Résumé du projet en anglais d’une longueur d’environ 20 lignes soit 200 à 250 mots, ou environ 1200 caractères espaces non compris 3](#_30j0zll)***

***[3. Cahier des charges ou expression des besoins du projet 4](#_1fob9te)***

***[4. Gestion de projet (planning et suivi, environnement humain et technique, objectifs de qualité) 5](#_3znysh7)***

***[5. Spécifications fonctionnelles du projet 6](#_2et92p0)***

***[6. Spécifications techniques du projet, élaborées par le candidat, y compris pour la sécurité 7](#_tyjcwt)***

***[7. Réalisations du candidat comportant les extraits de code les plus significatifs et en les argumentant, y compris pour la sécurité 8](#_3dy6vkm)***

***[8. Présentation du jeu d’essai élaboré par le candidat de la fonctionnalité la plus représentative (données en entrée, données attendues, données obtenues) 9](#_1t3h5sf)***

***[9. Description de la veille, effectuée par le candidat durant le projet, sur les vulnérabilités de sécurité 10](#_4d34og8)***

***[10. Description d’une situation de travail ayant nécessité une recherche et effectuée par le candidat durant le projet 11](#_2s8eyo1)***

# Liste des compétences du référentiel qui sont couvertes par le projet

1. Maquetter une application
2. Développer une interface utilisateur
3. Développer des composants d’accès aux données
4. Développer la partie front-end d’une interface utilisateur
5. Développer la partie back-end d’une interface utilisateur
6. Concevoir une base de données
7. Mettre en place une base de données
8. Développer des composants dans le langage d’une base de données
9. Concevoir une application
10. Développer des composants métiers
11. Construire une application organisée en couches
12. Développer une application mobile
13. Préparer et exécuter les plans de tests d’une application
14. Préparer et exécuter le déploiement d’une application

2. Résumé du projet en anglais

# 

SCI Cansell Real Estate Web Application offers features such as personalized search queries for buying, selling, or renting properties, as well as the ability for users to upload photos of their properties. For this application, I have implemented a form for accurate property valuation. Additionally, there is a dedicated space to showcase renovations done on previous properties.

I have created a mockup with five pages: Home, Search, Renovation, Estimation, and Image Upload. Each page serves a specific function.

For database modeling, I have used the MERISE approach, creating six fundamental tables, including roles, property photos, registered users, property valuations, categories, and property owners' data.

The use case diagram illustrates the interactions between application actors such as users, real estate agents, and administrators. Each actor plays an essential role in the functioning and interaction with the database.

Regarding the application development, I have utilized the Spring Boot framework for the backend, leveraging its features like dependency injection, transaction management, and integrated security. For the frontend, I have employed Angular, a TypeScript-based framework that allows for robust architecture and dynamic user interface creation.

I have implemented Spring Security dependencies and configured URL access, restricting certain functionalities based on user roles. In Angular, I have used routes and guards to control component access and implemented CRUD functionality to keep user and property data up-to-date.

# 3. Cahier des charges ou expression des besoins du projet

**a. Présentation de l'entreprise**

SCI Cansell est une entreprise familiale leader dans le secteur immobilier avec une remarquable expérience de plus de 50 ans, au cours desquels elle s'est distinguée en offrant des services de qualité et de confiance. Son principal engagement réside dans la satisfaction des besoins de ses clients en leur fournissant des solutions efficaces et adaptées à leurs exigences sur le marché immobilier dynamique.

L'entreprise s'est adaptée aux changements technologiques et cherche constamment à favoriser la rencontre entre propriétaires et acheteurs via des canaux numériques. Ils reconnaissent qu'à l'ère numérique, il est essentiel d'offrir aux clients le confort et l'accessibilité que procurent les plateformes en ligne. C'est pourquoi ils ont souhaité développer une application Web immobilière permettant aux utilisateurs d'explorer et de sélectionner des propriétés de manière pratique et efficace.

Ils ne se limitent pas seulement à présenter des propriétés aux clients, mais ils jouent également un rôle de conseil complet. Ils comprennent que le processus d'achat ou de vente d'une propriété implique des aspects légaux, financiers, fiscaux et contractuels qui peuvent être complexes et déroutants pour les parties impliquées. Ils s'engagent donc à fournir à leurs clients le soutien nécessaire dans ces domaines, en collaborant étroitement avec des professionnels du secteur tels que des notaires, des institutions financières et des experts en fiscalité et en contrats. De cette manière, ils garantissent que les clients disposent de toutes les informations et du soutien nécessaires pour prendre des décisions éclairées et sûres dans leurs transactions immobilières.

En ce qui concerne les clients, SCI Cansell se consacre à écouter attentivement leurs besoins et leurs exigences. Ils reconnaissent que chaque client a des exigences et des préférences uniques lorsqu'il recherche une propriété. C'est pourquoi ils s'efforcent de comprendre en profondeur les attentes de leurs clients et de leur présenter des propriétés qui correspondent précisément à leur profil de recherche. De plus, pour offrir encore plus de confiance et de sécurité, ils effectuent des évaluations et des inspections légales avant l'achat. Cela permet aux clients d'avoir une vision claire et complète des propriétés qu'ils envisagent, en s'assurant que leur investissement est judicieux et étayé par des informations fiables.

**b. Présentation du projet**

L'idée principale est de développer une application web immobilière offrant aux utilisateurs une expérience interactive et participative concernant leurs biens immobiliers. L'objectif est que les utilisateurs se sentent à l'aise et puissent avoir un meilleur contrôle sur l'information et la présentation visuelle de leurs propriétés.

Avec cette application, les utilisateurs pourront interagir plus activement avec les données de leurs biens immobiliers. Ils auront non seulement accès à des informations détaillées telles que les caractéristiques, l'emplacement et la description, mais pourront également contribuer avec leurs propres images des espaces qu'ils souhaitent montrer. Cependant, afin de garantir la qualité et la cohérence des images, celles-ci devront toutes passer par un processus de validation de la part de l'agence immobilière. Cela assurera que les photographies présentées soient attrayantes et précises, offrant une représentation fidèle de chaque propriété.

De plus, un service d'évaluation en ligne des biens immobiliers a été intégré. Ce service permettra aux utilisateurs d'obtenir une estimation du prix de leurs propriétés sans avoir besoin d'interagir directement avec un agent immobilier. Grâce à un agent immobilier spécialisé, les utilisateurs pourront fournir des informations pertinentes sur leurs biens et recevoir une évaluation précise et fiable. Cette fonctionnalité leur offrira une perspective claire de la valeur de leurs propriétés et les aidera à prendre des décisions éclairées concernant la vente, la location ou l'achat de biens immobiliers.

Il est important de souligner que pour accéder tant au service d'évaluation qu'à l'option de téléchargement d'images, les utilisateurs devront être enregistrés sur l'application web. L'inscription leur permettra de profiter pleinement de ces avantages et leur offrira une expérience personnalisée. De plus, dans un souci de sécurité et de protection des données, des mesures de sécurité ont été mises en place pour garantir que l'accès et l'utilisation de la plateforme soient sécurisés et confidentiels.

Cette application web immobilière vise à offrir aux utilisateurs une plateforme interactive et participative, où ils pourront interagir avec les informations de leurs biens immobiliers et jouer un rôle actif dans leur présentation visuelle. Avec des services d'évaluation en ligne et la possibilité de télécharger des images validées, l'idée est de fournir aux utilisateurs des outils qui les aideront à prendre des décisions intelligentes et bien fondées sur le marché immobilier. Tout cela, dans un environnement sûr et fiable qui favorise la confiance et la satisfaction des clients/utilisateurs."

**c. Fonctionnalités**

L'application web de l'agence immobilière a pour objectif principal de faciliter aux utilisateurs la recherche, la publication et l'évaluation de biens immobiliers de manière efficace et pratique. Voici les principales fonctionnalités de l'application :

Recherche de biens immobiliers : Les utilisateurs peuvent effectuer des recherches en fonction de leurs besoins spécifiques, que ce soit pour l'achat, la vente ou la location de propriétés. L'application propose des options de filtrage par catégorie (type de propriété, emplacement, prix, etc.) ainsi que par ID de propriété, ce qui permet aux utilisateurs de trouver facilement les biens qui correspondent à leurs préférences.

Publication d'images de biens immobiliers : Les utilisateurs ont la possibilité de publier leurs propres photos de manière simple et efficace. Ils peuvent télécharger des photographies de la propriété en suivant un processus rigoureux d'approbation de la part de l'agence immobilière. Cela permet aux utilisateurs de participer activement à la présentation visuelle de leur propriété, contribuant ainsi à la présenter de manière attrayante et précise.

Évaluation de biens immobiliers : L'application propose un service d'évaluation en ligne qui permet aux utilisateurs de connaître la valeur estimée de leurs biens immobiliers. Les utilisateurs peuvent saisir les informations pertinentes sur la propriété et recevoir une analyse détaillée et précise de son évaluation. Cette fonctionnalité offre aux utilisateurs un outil utile pour prendre des décisions éclairées concernant l'achat, la vente ou la location de biens immobiliers.

Rénovations de propriétés : L'application dispose d'un espace dédié pour présenter les rénovations réalisées sur différentes propriétés. Cela permet aux utilisateurs d'avoir une vision claire du potentiel d'amélioration et de la qualité des propositions immobilières. Cette fonctionnalité offre aux utilisateurs la possibilité d'explorer les transformations réalisées sur des biens immobiliers précédents, ce qui peut influencer leur décision d'achat ou de location.

Gestion des utilisateurs : L'application permet aux utilisateurs de s'inscrire et d'accéder à leur compte personnalisé. À travers leur compte, les utilisateurs peuvent effectuer des actions telles que télécharger des photos de biens immobiliers, demander des évaluations, mettre à jour des informations personnelles et gérer leurs propriétés dans l'application (il convient de préciser que les informations sur la propriété ne peuvent être mises à jour que par l'agence immobilière).

Rôles des utilisateurs : L'application différencie les rôles des utilisateurs, tels que l'administrateur de l'application. Chaque rôle a des privilèges et des fonctionnalités différentes dans l'application. Par exemple, les administrateurs peuvent publier des propriétés, les utilisateurs ne peuvent que rechercher et contacter un agent, tandis que l'utilisateur enregistré peut également télécharger des images d'une propriété qui lui appartient et/ou faire une évaluation.

Interaction avec les agents immobiliers : Les utilisateurs peuvent interagir avec les agents immobiliers à travers l'application pour obtenir des conseils personnalisés et des réponses à leurs questions. Cette fonctionnalité offre un canal de communication direct entre les utilisateurs et les experts immobiliers, améliorant ainsi l'expérience de l'utilisateur et offrant un plus grand niveau de confiance.

L'application web de l'agence immobilière offre aux utilisateurs une plateforme complète et conviviale pour rechercher, publier et évaluer. Elle permet aux utilisateurs de participer activement à la présentation visuelle de leurs propriétés et d'obtenir des évaluations précises. Avec des fonctionnalités supplémentaires telles que la visualisation des rénovations et l'interaction avec les agents immobiliers, l'application offre une expérience globale qui facilite la prise de décisions sur le marché immobilier."

**d. Possibles restrictions du projet d'application web immobilière**

Contrainte de temps : En raison d'une date limite, il est possible que le développement web ne puisse pas être entièrement achevé avant cette date.

Contrainte des ressources humaines : Les ressources humaines sont d'une importance capitale, car je ne dispose pas de la collaboration d'autres personnes pouvant m'aider en cas de doutes techniques ou ayant l'expérience des outils que j'utilise pour mener à bien le développement de l'application. Dans ce projet, je travaille de manière individuelle.

Contraintes de portée : Le projet a une portée définie avec des caractéristiques et des fonctionnalités spécifiques qui doivent être mises en œuvre dans l'application web.

Contraintes de sécurité : Étant donné que l'application stocke des informations sur les utilisateurs et les propriétés, il est impératif de respecter rigoureusement les normes de sécurité et de confidentialité. Des mesures doivent être prises pour protéger les informations confidentielles et se conformer aux réglementations en vigueur.

Ces restrictions sont des considérations importantes qui doivent être prises en compte lors du développement de l'application web immobilière

# 4. Gestion de projet (planning et suivi, environnement humain et technique, objectifs de qualité)

Étant donné que je travaille seule, j'applique les principes d'organisation et de suivi pour maximiser l'efficacité et m'assurer d'atteindre les objectifs fixés.

Pour commencer, j'ai établi les délais pour chaque tâche à réaliser, tels que la création du modèle de base de données, la conception et le développement. Fondamentalement, je les ai divisées en trois étapes ou tâches principales, et certaines de ces tâches ont été subdivisées en sous-tâches.

Dans cette optique, j'ai défini les principales fonctionnalités à mettre en œuvre, telles que la connexion et l'inscription de l'utilisateur, ainsi que la sécurité des données pour compléter l'interface utilisateur.

Certaines sous-tâches plus petites incluent le développement de la page d'accueil avec les informations de l'entreprise, la mise en place de la page de recherche et l'intégration des données pour charger toutes les informations de la page.

J'ai fixé des dates pour chaque tâche, comme la conception UX, le développement des pages, la connexion, les enregistrements, les validations, etc. Tout cela est planifié avec une date de début et une date de fin indiquées dans l'outil Trello, où chaque tâche a un temps spécifique pour être réalisée. Bien sûr, tout est sujet à des changements en cas d'imprévus.

J'ai également défini mes propres horaires de travail, en travaillant en continu du lundi au vendredi.

**Outils utilisés :**

Certaines des outils que j'ai utilisés sont :

**Trello :** J'utilise Trello pour suivre les tâches que je dois effectuer tout au long des semaines. J'attribue un temps estimé à chaque tâche pour planifier mon travail de manière efficace.

**GitHub :** J'utilise GitHub pour contrôler les versions de mes projets. Cela me permet de suivre les changements effectués pendant le développement du programme et de maintenir un historique actualisé des modifications.

**IntelliJ IDEA :** Travailler dans l'environnement IntelliJ IDEA a amélioré ma productivité dans le développement. Cet outil offre des fonctionnalités qui facilitent la création et la maintenance du projet, ce qui m'a permis d'optimiser mon flux de travail.

**Outils utilisés :**

Certaines des outils que j'ai utilisés sont :

Trello : J'utilise Trello pour suivre les tâches que je dois effectuer tout au long des semaines. J'attribue un temps estimé à chaque tâche pour planifier mon travail de manière efficace.

GitHub : J'utilise GitHub pour contrôler les versions de mes projets. Cela me permet de suivre les changements effectués pendant le développement du programme et de maintenir un historique actualisé des modifications.

IntelliJ IDEA : Travailler dans l'environnement IntelliJ IDEA a amélioré ma productivité dans le développement. Cet outil offre des fonctionnalités qui facilitent la création et la maintenance du projet, ce qui m'a permis d'optimiser mon flux de travail.

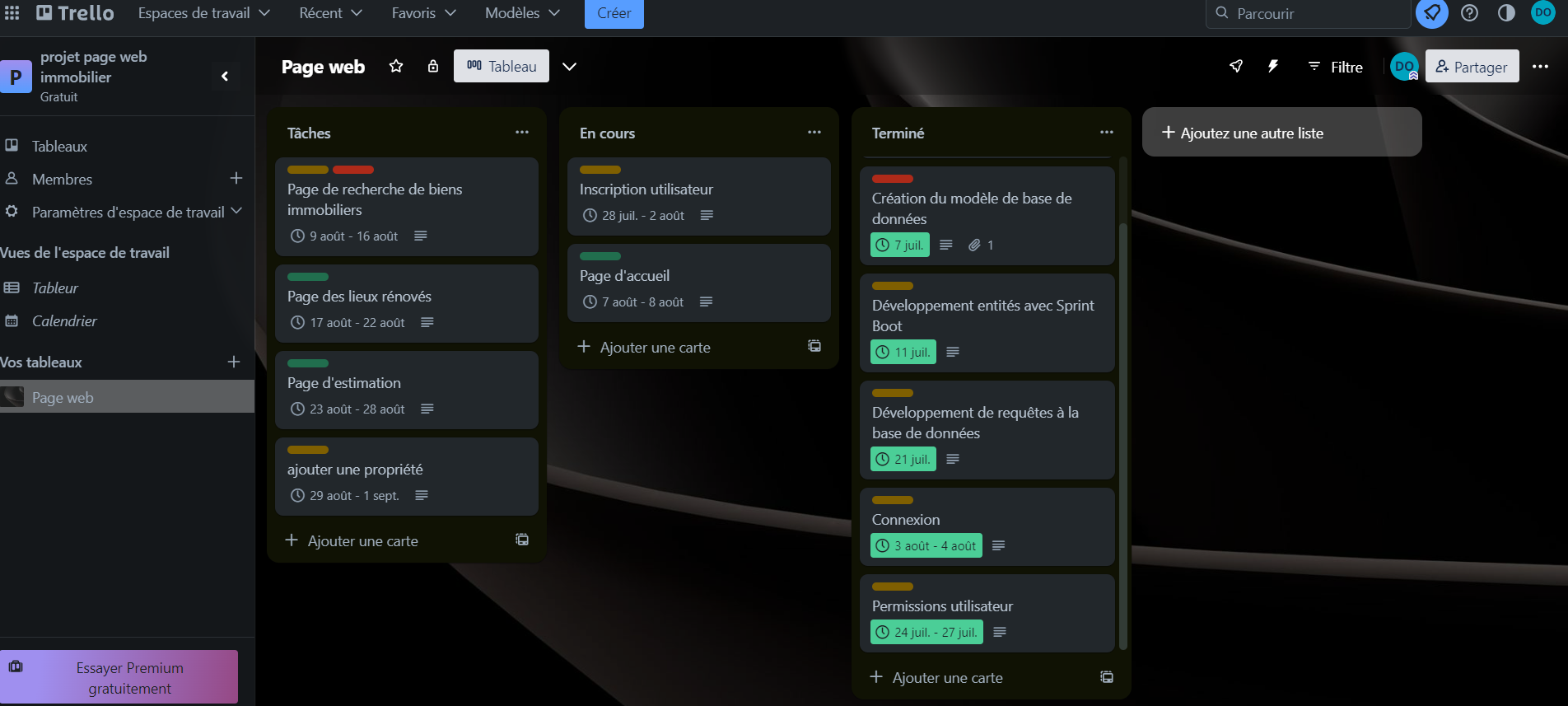
Changements de vision du client pendant le projet et adaptations

Tout au long de tout projet, il est compréhensible que le client, dans ce cas, l'agence immobilière, puisse demander des changements à un moment donné. Ces changements doivent être pris en compte dans le temps estimé pour le développement, le cas échéant. Pour cette raison, des temps supplémentaires ont été alloués à certaines tâches afin de pouvoir répondre de manière appropriée aux besoins du client.

Si le client souhaite ajouter de nouvelles fonctionnalités, il sera nécessaire de mettre en place de nouveaux changements dans le plan et les exigences pour inclure ces fonctionnalités. Il est important de souligner que ces changements sont effectués en tenant compte des temps de travail déjà établis.

La flexibilité pour s'adapter aux changements est essentielle dans le processus de développement du projet. Je dois être prêt(e) à ajuster le plan et les tâches en fonction des exigences du client. Cela implique d'évaluer l'impact des changements proposés, tant en termes de temps que de ressources, et de s'assurer que le projet puisse continuer à avancer de manière efficace et en ligne avec les attentes du client.

Ci-dessous, je vous présente une image de ces tâches

Figure #1->

# 5. Spécifications fonctionnelles du projet

1. **Cas d'utilisation : Modifier une propriété**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Administrateur*** | |
| ***Préconditions*** | ***accéder à l'application***  ***s'identifier*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Accéder à la page de recherche***  ***\* Sélectionner une propriété immobilière***  ***\* Modifier la surface de la propriété immobilière*** | ***permissions d'administrateur*** |
|
|
| ***Postconditions*** | ***Confirmer modifier le bien immobilier*** | |
|

1. **Cas d'utilisation : Valider la valeur d'un bien immobilier"**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Utilisateur*** | |
| ***Préconditions*** | ***accéder à l'application***  ***s'inscrire*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Aller sur la page de validation***  ***\* Remplir les champs avec les informations requises***  ***\* Envoyer*** | ***Vous devez être inscrit en tant qu'utilisateur*** |
|
|
| ***Postconditions*** | ***Attendre la réponse de agencent immobilière par e-mail*** | |
|

1. **Cas d'utilisation : Télécharger des photos d'une propriété"**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Utilisateur*** | |
| ***Préconditions*** | ***accéder à l'application***  ***s'inscrire*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Aller à la page de recherche.***  ***\* Sélectionner la propriété ou la rechercher par son ID.***  ***\* Cliquer sur le bouton "Télécharger des photos".***  ***\* Envoyer.*** | ***Vous devez être inscrit en tant qu'utilisateur*** |
|
|
|
|
| ***Postconditions*** | ***Attendre la validation des images par l'agence immobilière*** | |
|

1. **Cas d'utilisation : Rechercher une propriété"**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Utilisateur*** | |
| ***Préconditions*** | ***accéder à l'application*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Aller à la page de recherche.***  ***\* Rechercher par son ID / catégorie.***  ***\* Contacter un agent immobilier.*** |  |
|
|
|
|
| ***Postconditions*** | ***Attendre réponse de l'agence immobilière*** | |
|

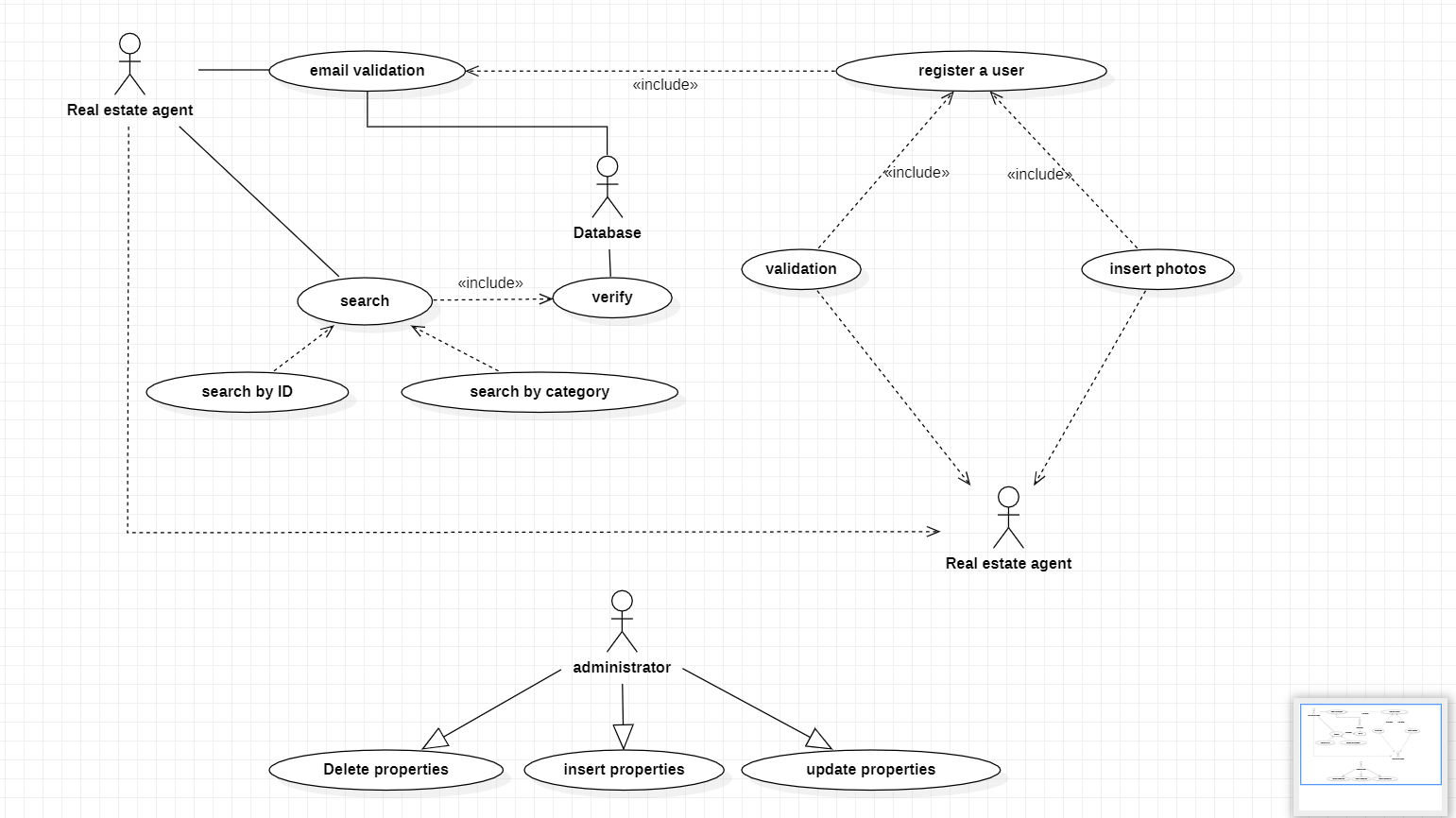
1. **Cas d'utilisation : Insérer une propriété"**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Administrateur*** | |
| ***Préconditions*** | ***accéder à l'application***  ***s'identifier*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Aller à la page de recherche.***  ***\* Rechercher par son ID / catégorie***  ***\* Sélectionner l'option "Insérer".***  ***\* Remplir les champs requis.***  ***\* Insérer les informations.*** | ***Avoir les permissions d'administrateur.*** |
|
|
|
|
| ***Postconditions*** | ***Notification d'insertion*** | |
|

1. **Cas d'utilisation : Supprimer une propriété"**

|  |  |  |
| --- | --- | --- |
| ***Acteurs*** | ***Administrateur*** | |
| ***Préconditions*** | ***accéder à l'application***  ***s'identifier*** | |
|
|
| ***Scenarios*** | ***Étapes*** | ***Règles de gestion*** |
| ***\* Aller à la page de recherche.***  ***\* Rechercher par son ID / catégorie***  ***\* Sélectionner la propriété***  ***\* Supprimer*** | ***Avoir les permissions d'administrateur.*** |
|
|
|
|
| ***Postconditions*** | ***Confirmer la suppression de la propriété*** | |
|

**Diagramme de classes**

****

**Dictionnaire de données**

**Role**

**role\_id: Identifiant unique du rôle. (Type de données : chaîne de caractères de longueur 50)**

**name\_role: Nom du rôle. (Type de données : chaîne de caractères de longueur 50)**

**PRIMARY KEY(role\_id): Clé primaire pour la table Role.**

**PropertyOwnerInformation**

**owner\_id: Identifiant unique du propriétaire. (Type de données : entier)**

**first\_name: Prénom du propriétaire. (Type de données : chaîne de caractères de longueur 50)**

**last\_name: Nom de famille du propriétaire. (Type de données : chaîne de caractères de longueur 50)**

**email: Adresse e-mail du propriétaire. (Type de données : chaîne de caractères de longueur 100)**

**telephone: Numéro de téléphone du propriétaire. (Type de données : chaîne de caractères de longueur 50)**

**address: Adresse du propriétaire. (Type de données : chaîne de caractères de longueur 50)**

**registered\_owner: Indicateur du propriétaire enregistré (Type de données : LOGIQUE)**

**PRIMARY KEY(owner\_id) : Clé primaire pour la table PropertyOwnerInformation.**

**CategoryContratct**

**category\_id: Identifiant unique de la catégorie. (Type de données : entier)**

**stateContract: Indique si un contrat a été établi avec le client. (Type de données : booléen)**

**PRIMARY KEY(category\_id): Clé primaire pour la table Category.**

**TypePhoto**

**type\_photo\_id: Identifiant unique du type de photo. (Type de données : entier)**

**type\_photo: Indique s'il s'agit d'une photo de propriété ou d'une photo de propriété rénovée (Type de données : VARCHAR(50)).**

**PRIMARY KEY(type\_photo\_id): Clé primaire pour la table TypePhoto.**

**PropertyState**

**property\_state\_Id: Identifiant unique de l'état de la propriété. (Type de données : entier)**

**descriptionState: Indique l'état de la propriété (en validation, approuvé) (Type de données : VARCHAR(100))**

**Property**

**property\_id: Identifiant unique de la propriété. (Type de données : entier)**

**real\_estate\_type: Type de bien immobilier. (Type de données : chaîne de caractères de longueur 50)**

**real\_estate\_age: Âge du bien immobilier. (Type de données : octet)**

**number\_showers: Nombre de douches. (Type de données : entier)**

**number\_bedrooms: Nombre de chambres. (Type de données : entier)**

**number\_rooms: Nombre total de chambres. (Type de données : entier)**

**needs\_renovation: Indicateur de la nécessité de rénovation (vrai/faux). (Type de données : booléen)**

**land\_surface: Superficie du terrain. (Type de données : décimal(15,2))**

**living\_room\_surface: Superficie du salon. (Type de données : décimal(15,2))**

**house\_surface: Superficie de la maison. (Type de données : décimal(15,2))**

**parking: Indicateur de présence d'un parking (vrai/faux). (Type de données : booléen)**

**cellar: Indicateur de présence d'une cave (vrai/faux). (Type de données : booléen)**

**cellar\_surface: Superficie de la cave. (Type de données : décimal(15,2))**

**balcony: Indicateur de présence d'un balcon (vrai/faux). (Type de données : booléen)**

**balcony\_surface: Superficie du balcon. (Type de données : décimal(15,2))**

**pool: Indicateur de présence d'une piscine (vrai/faux). (Type de données : booléen)**

**pool\_exposure: Exposition de la piscine. (Type de données : chaîne de caractères de longueur 50)**

**house\_view: Vue de la maison. (Type de données : chaîne de caractères de longueur 50)**

**house\_quality: Qualité de la maison. (Type de données : chaîne de caractères de longueur 50)**

**semi\_detached: Indicateur de maison mitoyenne (vrai/faux). (Type de données : booléen)**

**environment: Environnement de la propriété. (Type de données : chaîne de caractères de longueur 50)**

**validation\_state: État de validation. (Type de données : décimal(15,2))**

**description: Description de la propriété. (Type de données : chaîne de caractères de longueur 500)**

**price: Prix de la propriété. (Type de données : décimal(15,2))**

**category\_id: ID de catégorie liée (FK). (Type de données : entier, NON NULL)**

**owner\_id: ID du propriétaire lié (FK). (Type de données : entier, NON NULL)**

**PRIMARY KEY(property\_id): Clé primaire pour la table Property.**

**FOREIGN KEY(contract\_id): Clé étrangère faisant référence à la colonne contract\_id de la table CategoryContract.**

**FOREIGN KEY(validation\_state\_Id): Clé étrangère faisant référence à la colonne validation\_state\_Id de la table validationState.**

**Users**

**user\_id: Identifiant unique de l'utilisateur. (Type de données : entier)**

**first\_name: Prénom de l'utilisateur. (Type de données : chaîne de caractères de longueur 100)**

**last\_name: Nom de famille de l'utilisateur. (Type de données : chaîne de caractères de longueur 100)**

**telephone: Numéro de téléphone de l'utilisateur. (Type de données : chaîne de caractères de longueur 50)**

**address: Adresse de l'utilisateur. (Type de données : chaîne de caractères de longueur 250)**

**email: Adresse e-mail de l'utilisateur. (Type de données : chaîne de caractères de longueur 100)**

**password: Mot de passe de l'utilisateur. (Type de données : chaîne de caractères de longueur 100)**

**property\_id: ID de propriété liée (FK). (Type de données : entier, NON NULL)**

**role\_id: ID de rôle lié (FK). (Type de données : chaîne de caractères de longueur 50, NON NULL)**

**PRIMARY KEY(user\_id): Clé primaire pour la table Users.**

**FOREIGN KEY(property\_id): Clé étrangère faisant référence à la colonne property\_id de la table Property.**

**FOREIGN KEY(role\_id): Clé étrangère faisant référence à la colonne role\_id de la table Role.**

**Photo**

**id\_photo: Identifiant unique de la photo. (Type de données : entier)**

**jpg\_photo: Chemin de la photo au format JPG. (Type de données : chaîne de caractères de longueur 250)**

**user\_id: ID d'utilisateur lié (FK). (Type de données : entier, NON NULL)**

**property\_id: ID de propriété liée (FK). (Type de données : entier, NON NULL)**

**PRIMARY KEY(id\_photo): Clé primaire pour la table Photo.**

**FOREIGN KEY(user\_id): Clé étrangère faisant référence à la colonne user\_id de la table Users.**

**Belongs**

**FOREIGN KEY(photo\_id): Clé étrangère faisant référence à la colonne photo\_id de la table Photo.**

**FOREIGN KEY(property\_id): Clé étrangère faisant référence à la colonne property\_id de la table Property.**

**b. Méthodologie de modélisation MERISE.**

**Pour commencer, toutes les demandes du client ont été analysées, qui consistent essentiellement à créer des requêtes pour la recherche de biens immobiliers, que ce soit pour l'achat, la vente ou la location, ainsi qu'à permettre aux clients de publier facilement et efficacement des photos de leurs biens immobiliers. De plus, une fonctionnalité d'évaluation a été mise en place pour garantir une précision et un professionnalisme de la part des agents immobiliers. Pour cela, j'ai développé un formulaire gratuit où les spécialistes en immobilier évaluent chaque propriété avec minutie et fournissent la valeur de leur bien au client, parmi d'autres fonctionnalités.**

**Après avoir analysé les demandes du client, j'ai créé une base de données simple pour recueillir toutes les informations nécessaires à l'application. Plusieurs tables ont été conçues et reliées entre elles pour établir la logique de fonctionnement de l'application.**

**La table "Role" stocke les différents rôles que peuvent avoir les utilisateurs de l'application, tels que "administrateur" et "utilisateur". Il y a une relation de un à plusieurs entre la table "Role" et la table "Users", car un utilisateur peut avoir un seul rôle, mais un rôle peut être associé à plusieurs utilisateurs.**

**La table "User" contient des informations à la fois sur les utilisateurs enregistrés et sur les clients. La différence entre un client et un utilisateur enregistré est que le client a un contrat avec l'agence et seul l'administrateur peut enregistrer des informations relatives à ce client. Sauf si le client souhaite insérer ses propres images, auquel cas il doit changer son statut pour devenir un "utilisateur enregistré" avec contrat.**

**La table "Propertys" enregistre des informations sur les biens immobiliers. Cette table remplit deux fonctions : enregistrer toutes les informations nécessaires sur les biens immobiliers des clients, et permettre à un utilisateur enregistré de demander une évaluation de la valeur d'un bien immobilier. Un agent immobilier expert répondra ensuite à l'utilisateur en lui indiquant la valeur du bien. Il y a une relation de un à plusieurs entre la table "Users" et la table "Propertys", car un utilisateur peut avoir plusieurs propriétés, mais une propriété ne peut appartenir qu'à un seul utilisateur.**

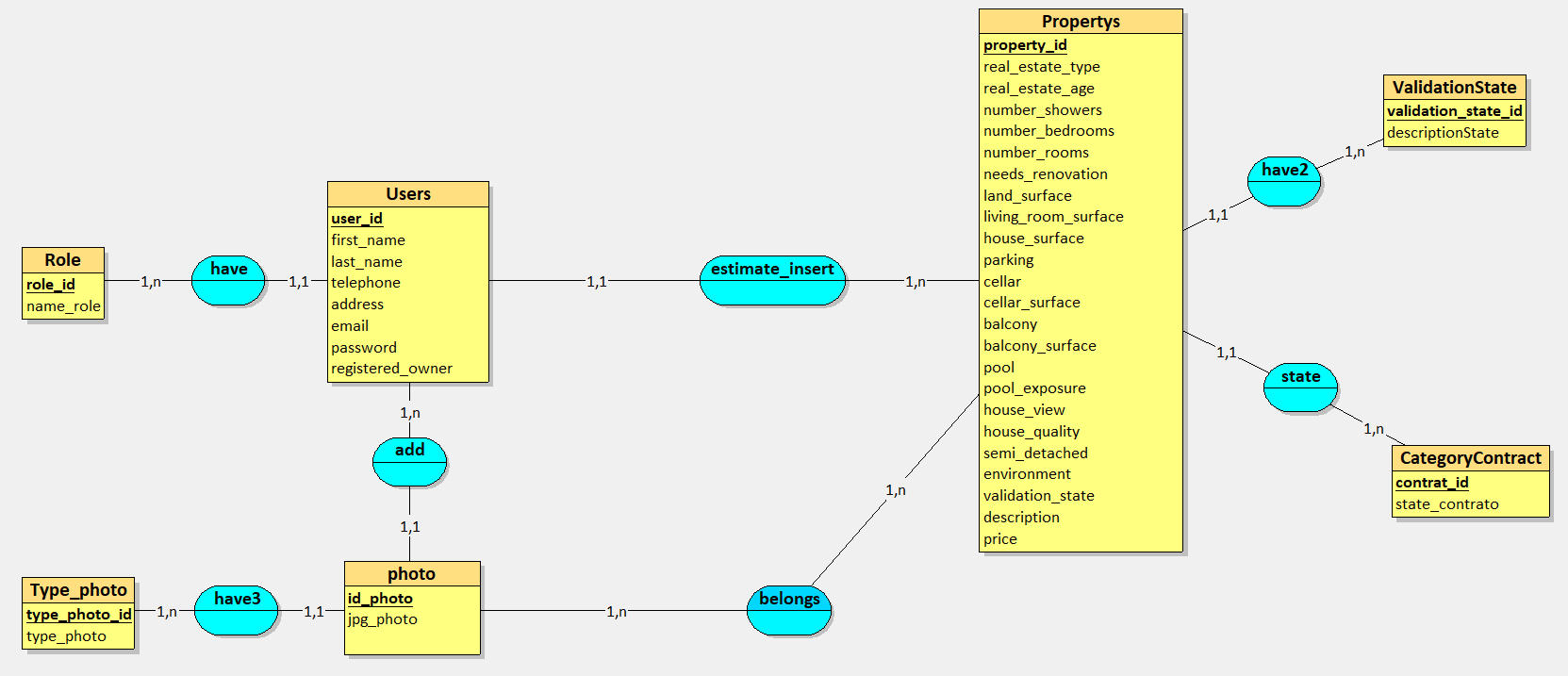
**La table "Category\_contract" est liée à la table "Propertys". Cette relation est de un à plusieurs, ce qui signifie qu'une propriété peut avoir un état de catégorie, mais un état de catégorie peut être attribué à plusieurs propriétés.**

**La table "Photo" est liée à la table "Users" et à la table "Propertys". Il y a une relation de un à plusieurs entre la table "User" et la table "Photos", car un utilisateur peut télécharger plusieurs photos. De plus, il y a une relation de un à plusieurs entre la table "Photo" et la table "Propertys", car une propriété peut avoir plusieurs photos et une photo peut appartenir à une seule propriété.**

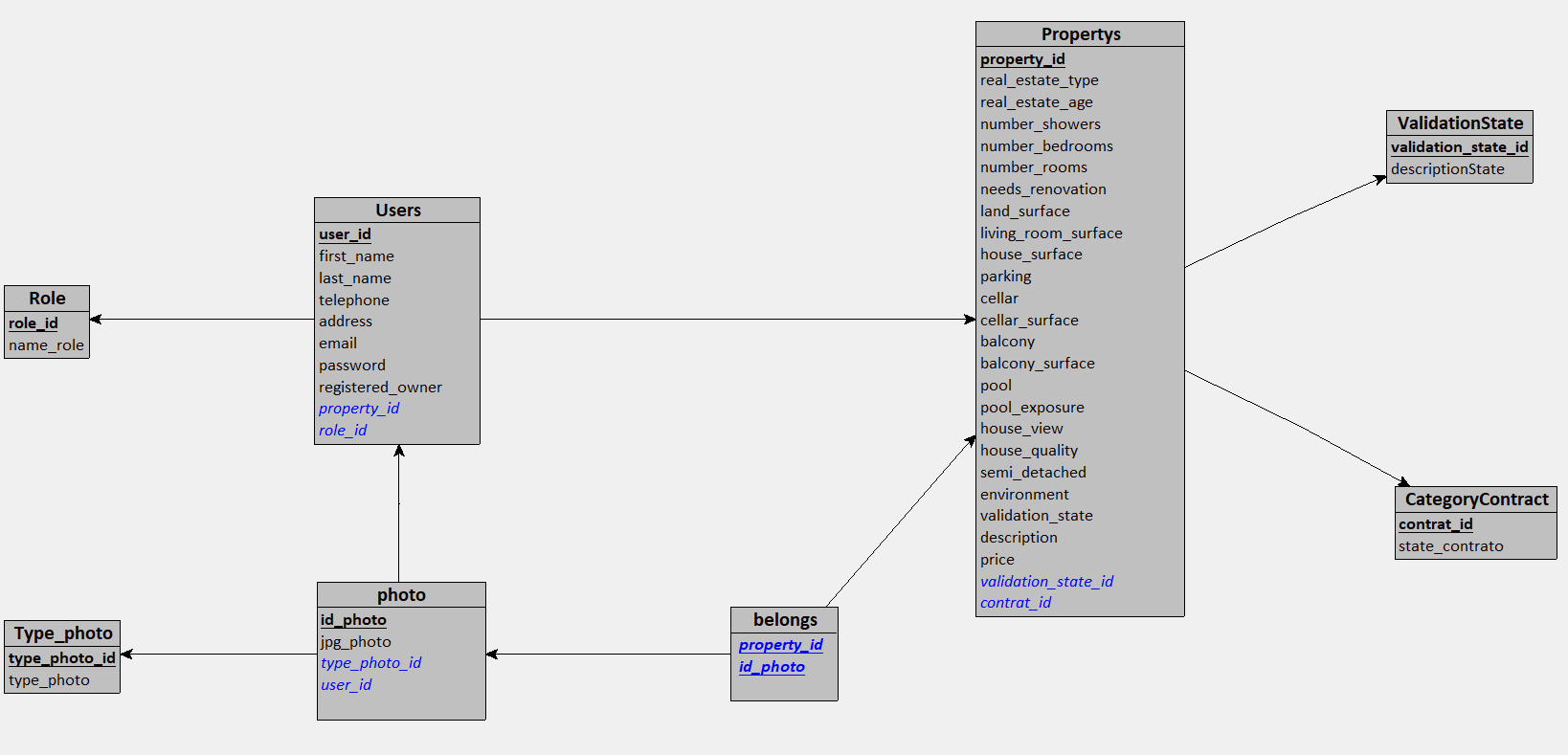
**La table "ValidationState" est directement liée à la table "Propertys", car elle permet de distinguer entre un client et un utilisateur enregistré qui valide sa propriété. Le client a déjà signé un contrat avec l'agence, tandis que l'utilisateur enregistré n'en a pas pour le moment.**

**J'ai appliqué une logique cohérente dans la conception de la base de données pour répondre aux besoins du client et établir les relations appropriées entre les entités. Cette structure permettra un meilleur contrôle et gestion des informations liées aux rôles des utilisateurs, aux propriétés, aux catégories et aux photos.**

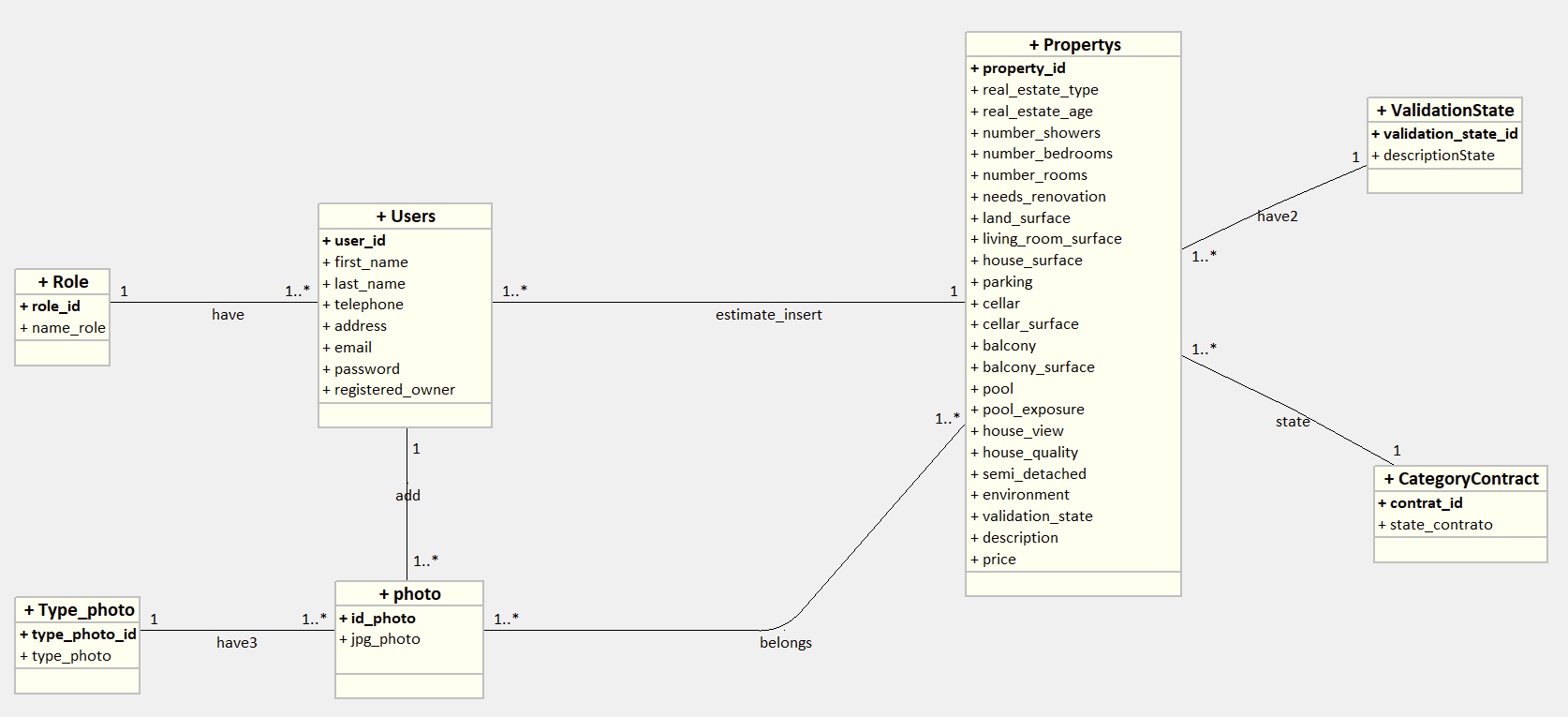
**Niveau Conceptuel MCD**

****

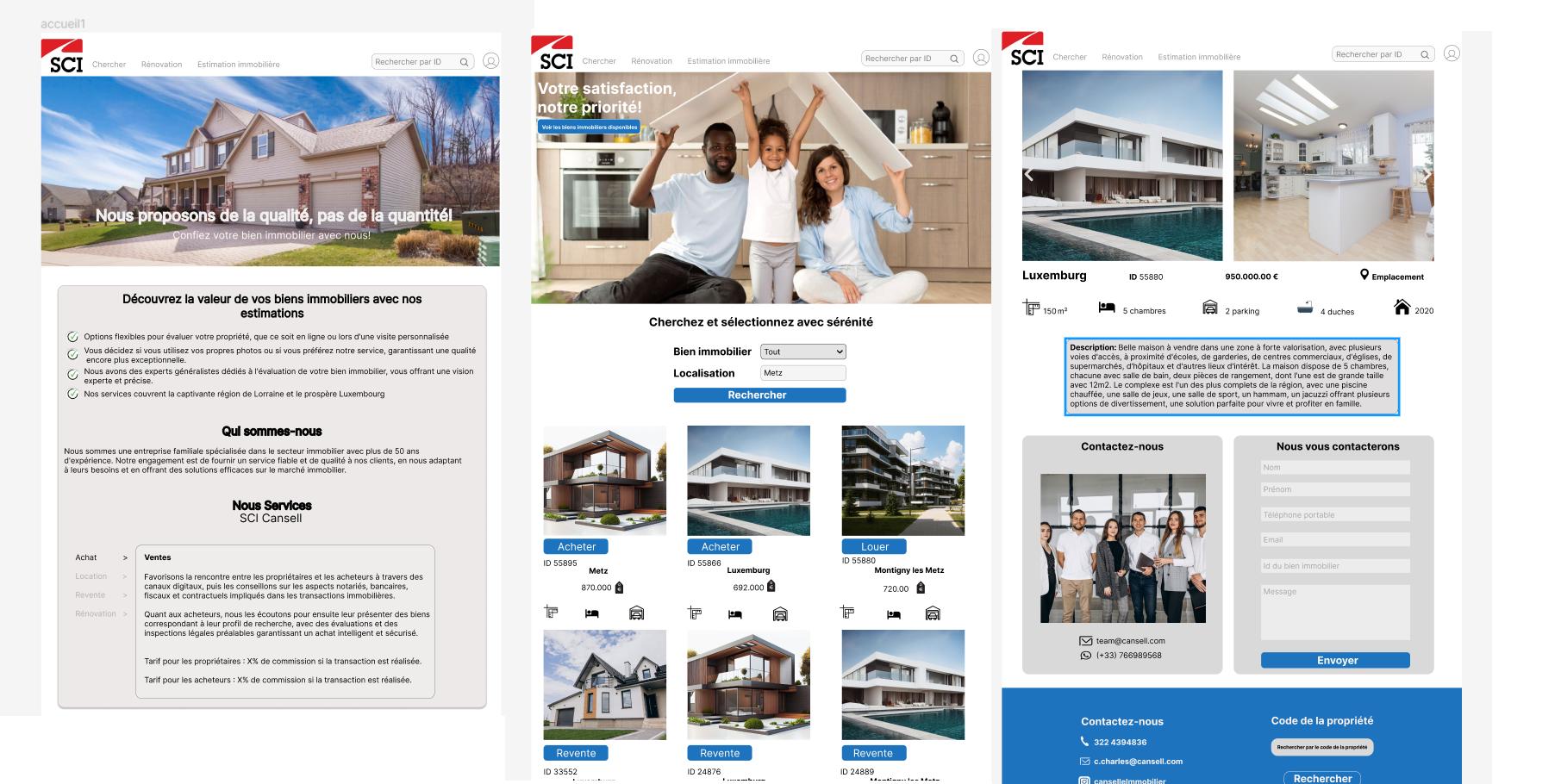
**Niveau Logique MLD**

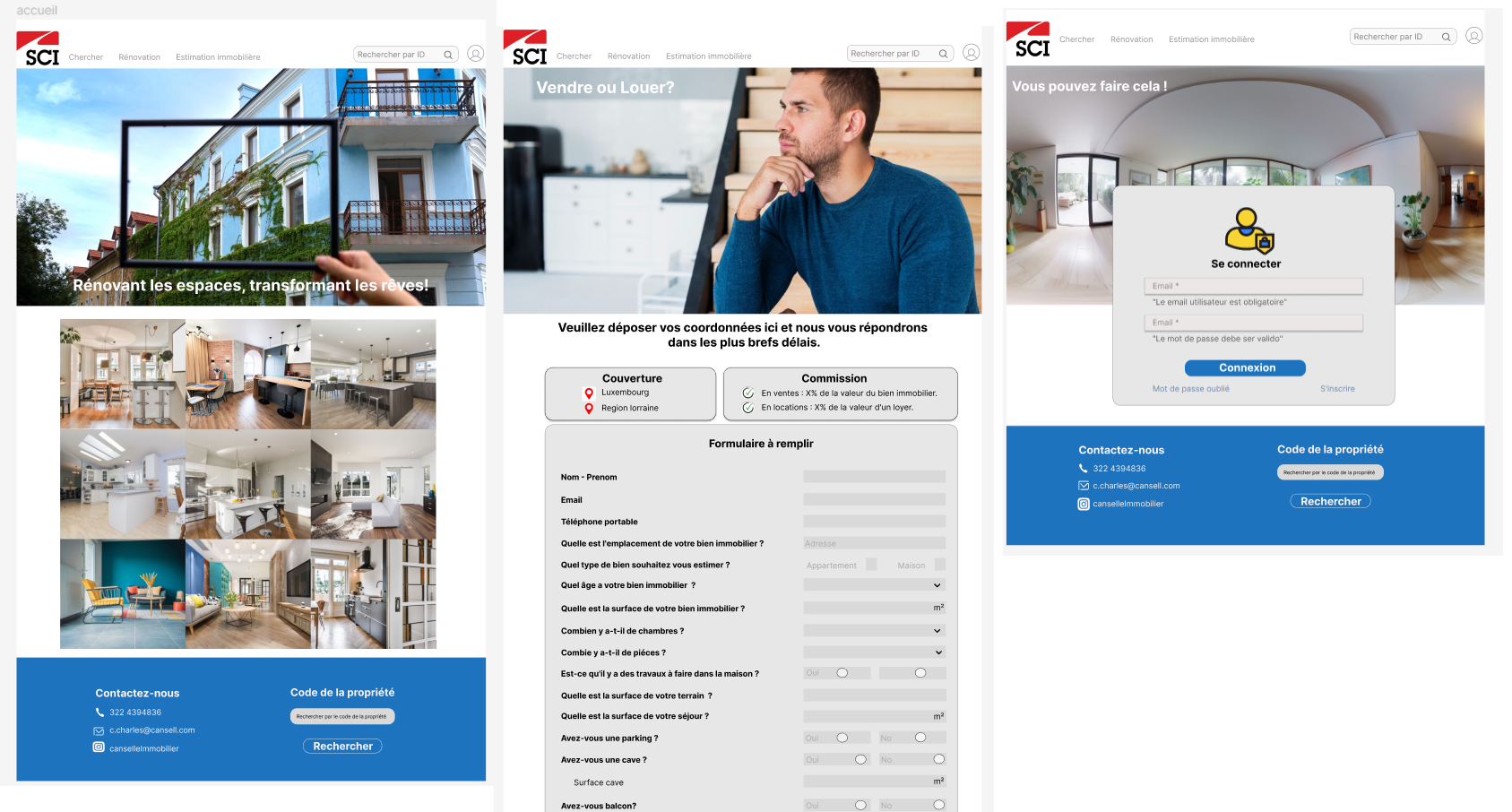
****

**Modélisation UML**

****

**c. Maquette ou conception UX**

****

****

**Le diseño du site Web a été développé en tenant compte des besoins spécifiques du client. Ci-dessous, vous trouverez des informations plus détaillées sur chacune des pages principales:**

**Page d'accueil : Cette page vise à fournir aux visiteurs une vue d'ensemble de l'agence immobilière. Elle inclut des informations pertinentes telles que l'histoire de l'entreprise, son expérience sur le marché et les politiques régissant ses services. De plus, la mission et les valeurs de l'entreprise sont mises en avant pour transmettre confiance et crédibilité.**

**Page de recherche : Sur cette page, les utilisateurs peuvent explorer et rechercher parmi toutes les propriétés disponibles. Ils ont un espace de recherche où ils peuvent saisir l'identifiant de la propriété ou sélectionner un type d'opération, comme la vente, la location ou l'achat. De plus, il est possible de filtrer les résultats par emplacement géographique. Une fois qu'ils trouvent une propriété d'intérêt, ils peuvent accéder à une page dédiée à cette propriété, où des informations détaillées telles que des descriptions, des caractéristiques, des photos et des détails supplémentaires sont présentés.**

**Page de rénovation : Dans cette section, des exemples d'espaces qui ont été rénovés sont présentés. Des photos des propriétés après avoir été soumises à des processus de rénovation et d'amélioration sont montrées. Cela donne aux utilisateurs une idée claire de l'apparence des propriétés après la rénovation, ce qui peut être un facteur important dans leur prise de décision.**

**Page d'estimation : Cette page offre aux utilisateurs la possibilité d'accéder à un formulaire pour effectuer des estimations. Les champs du formulaire sont conçus pour recueillir des informations sur la propriété afin de fournir une réponse précise de sa valeur. Il convient de noter que les champs du formulaire ne sont activés que si l'utilisateur est inscrit sur la plateforme, permettant ainsi une plus grande personnalisation.**

**Page d'inscription : Sur cette page, un accès direct est fourni aux utilisateurs enregistrés pour se connecter à leurs comptes. Si un utilisateur n'est pas encore enregistré, des options claires et visibles lui sont proposées pour qu'il puisse créer un compte sur la plateforme. L'inscription implique de remplir un formulaire avec des informations de base telles que le nom, l'adresse, le téléphone et l'adresse e-mail. Une fois enregistré, l'utilisateur aura accès à des fonctionnalités telles que l'évaluation des biens immobiliers et l'insertion de photos.**

**Il est important de noter que toutes les pages de l'application Web comportent un pied de page contenant les coordonnées de l'agence immobilière, telles que le numéro de téléphone, l'adresse e-mail et les liens vers les réseaux sociaux. Cela permet aux utilisateurs de contacter facilement l'agence pour obtenir plus d'informations, résoudre des doutes ou poser des questions spécifiques.**

**En plus des espaces mentionnés ci-dessus, l'application Web inclut des fonctionnalités spécifiques pour les utilisateurs enregistrés et l'administrateur. Les utilisateurs enregistrés ont accès à des boutons visibles leur permettant d'insérer des photos. D'autre part, l'administrateur dispose d'un ensemble complet d'outils lui permettant de gérer et de contrôler les informations des propriétés et des utilisateurs. Cela inclut la capacité de réaliser des modifications, des mises à jour et des suppressions d'informations.**

# 6. Spécifications techniques du projet, élaborées par le candidat, y compris pour la sécurité

**a. Choix des technologies**

Pour ce projet, j'ai sélectionné différentes technologies pour répondre aux exigences de développement. Les voici :

**Spring Boot :** C'est un framework Java qui facilite le développement d'applications web en fournissant une configuration automatique et des fonctionnalités intégrées. Avec Spring Boot, j'ai pu simplifier la mise en œuvre de l'architecture MVC (Modèle-Vue-Contrôleur) et bénéficier d'une gestion efficace des requêtes HTTP, de la sécurité et des exceptions.

**ORM Hibernate et JPA :** Hibernate est un framework de mappage objet-relationnel qui facilite la manipulation des données dans une base de données relationnelle. JPA (Java Persistence API) est une spécification qui définit une interface commune pour interagir avec la base de données. Ensemble, Hibernate et JPA me permettent d'effectuer des opérations de création, lecture, mise à jour et suppression (CRUD) sur les entités de manière simplifiée.

**Angular :** Il me permet d'établir facilement une architecture Modèle-Vue-Contrôleur (MVC), ce qui facilite la séparation des composants et améliore la maintenabilité du code. De plus, Angular offre un large éventail de fonctionnalités et d'outils qui aident au développement d'applications web de haute qualité.

**b. Choix de la base de données**

J'ai choisi d'utiliser une base de données relationnelle MariaDB pour ce projet. Ce choix est basé sur la structure des données et les besoins de stockage. Une base de données relationnelle permet de gérer efficacement les relations entre les entités et offre des fonctionnalités avancées pour l'intégrité des données et la gestion des transactions.

**c. Solution Back-end**

La solution back-end repose sur l'utilisation de Spring Boot, Hibernate et JPA. Ces technologies me permettent de développer une couche de logique métier robuste et efficace pour l'application. Spring Boot facilite la configuration et le déploiement du serveur, tandis que Hibernate et JPA simplifient l'interaction avec la base de données et la manipulation des données.

**Respect de l'architecture MVC :** Dans la conception de l'application, j'ai suivi le modèle d'architecture MVC (Modèle-Vue-Contrôleur), où le modèle représente les données et la logique métier, la vue s'occupe de l'interface utilisateur et le contrôleur gère les interactions entre le modèle et la vue.

**Tester notre API :** Pendant le développement, il est important de réaliser des tests exhaustifs de l'application pour assurer son bon fonctionnement. Cela implique de réaliser des tests unitaires pour garantir que les fonctionnalités répondent aux exigences établies. À cette fin, j'ai créé quelques tests unitaires en utilisant des outils tels que POSTMAN.

**d. Solution Front-End**

Pour la partie front-end, j'ai utilisé le framework Angular, Angular Material Bootstrap et CSS. Cela me permet de développer une interface utilisateur moderne et réactive pour l'application web. J'ai suivi une architecture spécifique pour organiser et modulariser le code, comme indiqué précédemment, et j'utiliserai des services pour implémenter la logique de l'application et effectuer des appels vers le back-end.

**Sécurité :** La sécurité est une considération importante dans le développement de toute application. J'ai mis en place des mesures de sécurité telles que l'utilisation de l'authentification basée sur les tokens JWT (JSON Web Tokens). J'ai également utilisé des gardes d'authentification (AuthGuards) côté front-end pour contrôler l'accès à certaines fonctionnalités en fonction des rôles et des permissions des utilisateurs.

J'ai défini l'environnement de développement en utilisant Spring Boot, ce qui m'a permis de mapper les entités des tables dans la base de données. Cela inclut la création de tables, la définition des noms, des colonnes et des relations entre les entités. Grâce à des fonctionnalités telles que l'ORM (Mapping Objet-Relationnel) et la bibliothèque Hibernate, je peux générer automatiquement des requêtes. Par exemple, la méthode save() génère une requête d'insertion et findAll() renvoie une requête pour obtenir tous les enregistrements d'une table, entre autres.

En ce qui concerne le développement du back-end, j'ai suivi une approche détaillée dans la modélisation de la base de données et de ses relations, telles que One-to-Many, Many-to-One et Many-to-Many, en fonction des besoins spécifiques de ma base de données relationnelle. En utilisant Spring Boot, j'ai réussi à créer la base de données dans son intégralité, car cet outil me permet non seulement de créer les tables, mais aussi d'intégrer le modèle de manière efficace.

Ensuite, j'ai créé des interfaces via des référentiels ou DAO (Data Access Object) qui facilitent la communication avec la base de données (en utilisant MariaDB). Ces référentiels contiennent les méthodes nécessaires pour effectuer des opérations de lecture, écriture, mise à jour et suppression (CRUD) dans la base de données. Cela m'a permis d'établir efficacement les requêtes et les transactions nécessaires dans des classes concrètes.

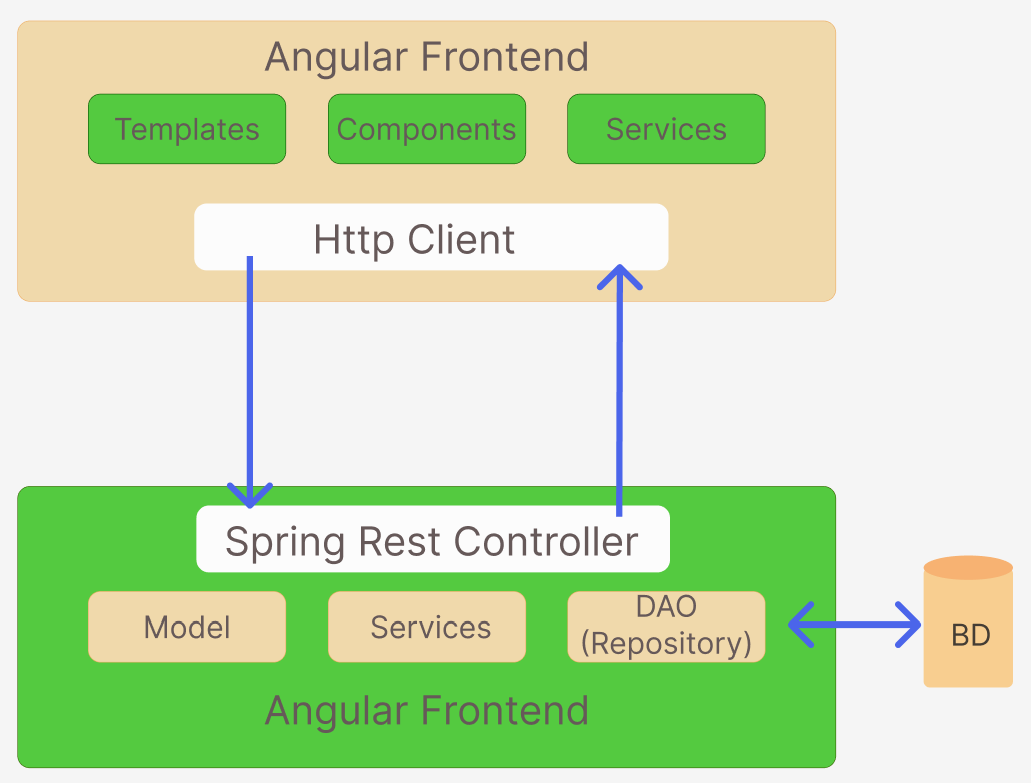
De plus, Spring Boot facilite les opérations de base du CRUD (Créer, Lire, Mettre à jour, Supprimer), simplifiant l'interaction avec la base de données et évitant la nécessité d'écrire manuellement des requêtes SQL. Certains des méthodes que j'ai utilisées sont save(), findById(), findAll(), update() et delete().

Pour intégrer le client web de l'agence immobilière, j'ai utilisé la classe 'HttpClient' d'Angular. Cette classe m'a permis d'effectuer différentes requêtes HTTP telles que GET, POST, PUT et DELETE, facilitant ainsi la gestion des réponses du serveur. En utilisant cette fonctionnalité, j'ai pu établir la couche de communication nécessaire entre le frontend et le backend de l'application.

De plus, j'ai créé une page qui sert de vue principale de l'application. Dans cette vue, la logique de connexion et d'inscription des utilisateurs est mise en œuvre, ainsi que les pages de navigation. Pour améliorer la modularité et la réutilisation du code, j'ai utilisé des composants d'Angular pour mettre en œuvre la logique spécifique de chaque section de l'application.

D'autre part, j'ai utilisé un package de services dans Angular pour établir la communication avec le backend développé en Spring Boot. Ces services agissent comme des intermédiaires entre le frontend et le backend, permettant l'échange de données et la gestion des opérations nécessaires pour obtenir et envoyer des informations au serveur.

En ce qui concerne la conception, j'ai décidé d'utiliser CSS, Bootstrap et Angular Material. Ce sont des outils très populaires et faciles à utiliser qui offrent une large gamme de styles et de composants prédéfinis. Avec une grande quantité d'informations et de documentation disponibles, je peux rapidement mettre en œuvre des conceptions attrayantes et fonctionnelles dans mon application. De plus, ces outils me permettent de rendre l'application responsive, ce qui est essentiel de nos jours car les utilisateurs accèdent aux applications depuis une variété d'appareils.



Dans cette image, le comportement intégral de Spring Boot et Angular est représenté. Du côté du backend, nous observons le pattern MVC (Modèle-Vue-Contrôleur), qui se connecte à la base de données via le DAO ou le Répertoire. Depuis Angular, un client se connecte au backend via le HttpClient.

Lorsqu'une opération est souhaitée, par exemple, insérer un utilisateur, une méthode dans le service du frontend est appelée en utilisant les méthodes HTTP telles que POST, GET ou DELETE. Le service appelle le contrôleur via le HttpClient, établissant ainsi une connexion. À son tour, le contrôleur s'appuie sur un service qui communique avec le DAO par injection de dépendances.

Ensuite, le DAO se connecte à la base de données et reçoit une réponse, qui est ensuite renvoyée au service. Le service renvoie ensuite les informations au frontend (Angular), et le résultat est affiché dans un modèle.

Ainsi, le HttpClient permet d'établir une connexion entre le backend et le frontend, où le pattern MVC est utilisé pour organiser la logique de l'application et l'accès à la base de données.

# 7. Réalisations du candidat comportant les extraits de code les plus significatifs et en les argumentant, y compris pour la sécurité

Création d'une entité dans Spring Boot

Pour illustrer cet exemple, j'ai choisi l'entité "utilisateurs", qui représente les utilisateurs dans le système. La relation entre l'entité "utilisateurs" et l'entité "rôle" est une relation Many-to-One (plusieurs à un). Chaque utilisateur peut avoir un seul rôle, tandis que plusieurs utilisateurs peuvent avoir le même rôle. La relation est représentée par les annotations @ManyToOne et est basée sur les champs "role\_id" et "rôle" dans l'entité "utilisateur", qui font référence à l'entité "rôle".

Quant à l'entité "rôle", qui représente les rôles dans le système, cette entité n'a pas de relation directe avec d'autres entités dans le schéma fourni. Cependant, en raison de la relation Many-to-One mentionnée ci-dessus dans l'entité "utilisateurs", les rôles peuvent être associés à plusieurs utilisateurs via la clé étrangère "role\_id".

``` java

package com.cansell.web.model;

import lombok.Data;

import org.hibernate.annotations.GenericGenerator;

import javax.persistence.\*;

@Entity

@Data

@Table(name = "users", uniqueConstraints = @UniqueConstraint(columnNames = "email"))

public class Users {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY, generator = "native")

@GenericGenerator(name = "native", strategy = "native")

@Column(name = "user\_id")

private Long userId;

@Column(name = "first\_name")

private String firstName;

@Column(name = "last\_name")

private String lastName;

@Column(name = "phone")

private String phone;

@Column(name = "address")

private String address;

@Column(name = "email")

private String email;

@Column(name = "password")

private String password;

@Column(name = "registered\_owner")

private boolean registeredOwner;

@ManyToOne

@JoinColumn(name = "property\_id")

private Propertys propertyId;

@ManyToOne

@JoinColumn(name = "role\_id")

private Role role;

}

```