

4 MIN EAM

# Projet architecture Web

Enseignant : Mr DEKIMPE

The background of the ECAM logo features a stylized gear with a compass rose integrated into its upper section.

**ECAM**

**BRUSSELS ENGINEERING SCHOOL**

Matthias Léonard 20308  
2022/2023

## Table des matières

Introduction : .....	2
User stories : .....	2
MockUP : .....	3
Diagramme entité association : .....	5
Diagramme relationnel : .....	5
BackEnd : .....	5
Description des champs de la base de données : .....	6
Table appartements: .....	6
Table renters: .....	6
Tables repair_estimates : .....	6
Table tickets : .....	6
Description des tables et des relations entre tables : .....	6
ORM : .....	7
API : .....	7
Authentification : .....	8
FrontEnd Angular: .....	8
Architecture: .....	11
FrontEnd App mobile hybride Ionic: .....	12
Conclusion : .....	14

## Introduction :

Dans le cadre du cours de web architecture du cursus 4MIN il nous devons de réaliser un projet intégrant une API pour un développement front-end et back-end.

Le choix de mon projet est basé sur une agence immobilière, celle-ci qui doit pouvoir gérer sont portefeuille de biens immobilier mis à la location. La gestion des problèmes techniques et les locataires des biens.

Dans ce projet nous nous limiterons aux problèmes techniques associer des devis, associer des locataires à des biens. Il est bien évidemment possible d'approfondir ce projet notamment par la gestion des loyers et l'exercice comptable, la gestion de plusieurs propriétaires d'un bien, ces exemples ne sont pas exhaustifs.

Lien GitHub : <https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere>

## User stories :

User stories	Priorité
<b>Gestion des classes de biens</b>	
En tant qu'agence immobilière, je veux pouvoir ajouter un bien d'un propriétaire dans un lot	1
En tant qu'agence immobilière, je veux pouvoir supprimer un bien d'un lot	1
<b>Gestion d'un bien</b>	
En tant qu'agence immobilière, je veux pouvoir crée un bien	1
En tant qu'agence immobilière, je veux pouvoir supprimer un bien	1
En tant qu'agence immobilière, je veux pouvoir modifier un bien	3
<b>Gérer les locataires</b>	
En tant qu'agence immobilière, je veux pouvoir ajouter un locataire à un bien	4
En tant qu'agence immobilière, je veux pouvoir supprimer un locataire d'un bien	4
<b>Gérer les problèmes technique</b>	
En tant qu'agence immobilière, je veux pouvoir crée un ticket pour un problème technique	2
En tant qu'agence immobilière, je veux pouvoir supprimer un ticket	3
En tant qu'agence immobilière, je veux pouvoir lister les tickets	2
En tant qu'agence immobilière, je veux pouvoir associer un devis à un ticket	2

MockUP :

**LOTS immobilier**

Lot 1  
Propriétaire

Appartement

Supprimer

Lot 2  
Propriétaire

Appartement

Supprimer

Ajouter un lot

**Ajout d'un Lot**

Nom du propriétaire:  
Prénom :

Ajouter

**Appartement**

Appartement 1  
Valeur : 250 000€  
Loyer: 800€/mois

Locataire

Modifier

Prob  
tech

Supp

Appartement 2  
Valeur : 150 000€  
Loyer : 500€/mois

Locataire

Modifier

Prob  
tech

Supp

Ajouter un  
appartement

Locataire

Nom : Martin

Prénom : Jean-marc

Ajouter

Supprimer

ProblèmesTechnique

Ticket N°1

02/02/2023

Fuite d'eau

Lier un devis

Supprimer

Ticket N°2

02/02/2023

Fuite d'eau

Lier un devis

Supprimer

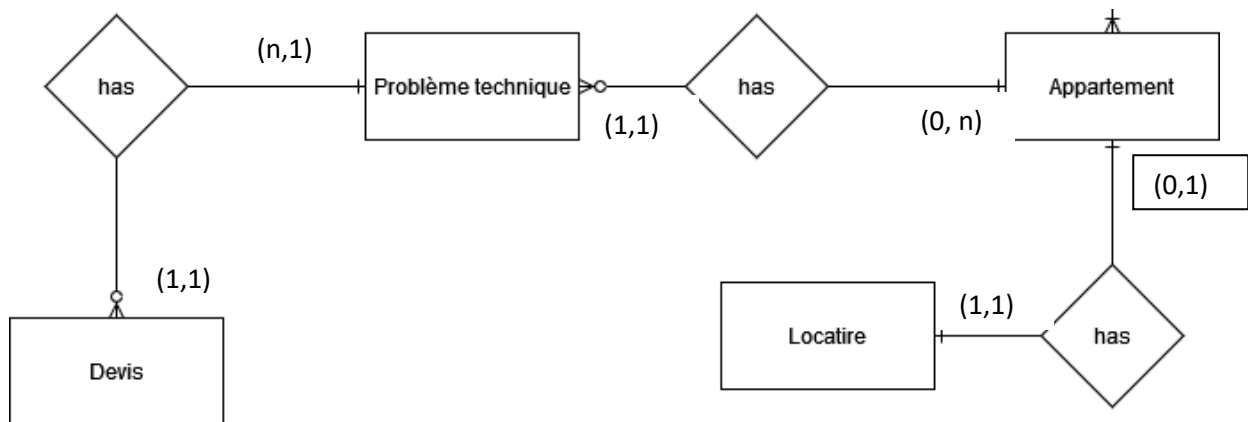
Crée un nouveau ticket

Devis

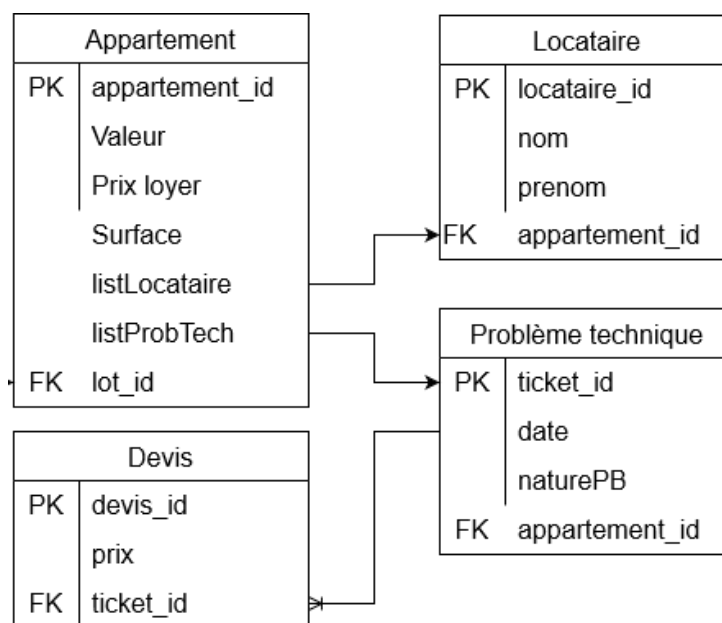
Prix du devis :

Ajouter

## Diagramme entité association :



## Diagramme relationnel :



## BackEnd :

Github : <https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/tree/main/WEB/BackEnd>

## Description des champs de la base de données :

### Table appartements:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra	Comments
◇ address	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ appart_Id	int		NO			select,insert,update,references	auto_increment	
◇ createdAt	datetime		NO			select,insert,update,references		
◇ description	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ nbRooms	int		NO			select,insert,update,references		
◇ price	int		NO			select,insert,update,references		
◇ surface	int		NO			select,insert,update,references		
◇ title	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ updatedAt	datetime		NO			select,insert,update,references		

### Table renters:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra	Comments
◇ appart_Id	int		NO			select,insert,update,references		
◇ createdAt	datetime		NO			select,insert,update,references		
◇ email	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ firstName	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ lastName	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ renter_Id	int		NO			select,insert,update,references	auto_increment	
◇ updatedAt	datetime		NO			select,insert,update,references		

### Tables repair\_estimates :

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra	Comments
◇ createdAt	datetime		NO			select,insert,update,references		
◇ description	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ estimate_Id	int		NO			select,insert,update,references	auto_increment	
◇ price	int		NO			select,insert,update,references		
◇ ticket_Id	int		NO			select,insert,update,references		
◇ title	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ updatedAt	datetime		NO			select,insert,update,references		

### Table tickets :

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra	Comments
◇ appart_Id	int		NO			select,insert,update,references		
◇ createdAt	datetime		NO			select,insert,update,references		
◇ description	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ status	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ ticket_Id	int		NO			select,insert,update,references	auto_increment	
◇ title	varchar(255)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references		
◇ updatedAt	datetime		NO			select,insert,update,references		

## Description des tables et des relations entre tables :

Table	Table	Type	Description
appartements	renters	1 à 1	Un appartement à un locataire et un locataire à un appartement.
appartements	tickets	0 à N	Un appartement peut avoir aucun ticket ou plusieurs tickets, un ticket appartient à un appartement.
tickets	repair_estimates	1 à N	Un ticket peut avoir plusieurs devis et un devis appartient à un ticket.

ORM :

<https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/blob/main/WEB/BackEnd/models/index.js>

API :

L'API est disponible en consultation sur le site de swaggerHub :

[https://app.swaggerhub.com/apis/LeTouristeDeLECAM/Agence\\_Immobiliere/1.0](https://app.swaggerhub.com/apis/LeTouristeDeLECAM/Agence_Immobiliere/1.0)

Ou sur GitHub sous format YAML : [https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/blob/main/WEB/API/API\\_estate\\_agency.yaml](https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/blob/main/WEB/API/API_estate_agency.yaml)

## Property Management

GET	/appartement	Get all appartements	✓ ↩
POST	/appartement	Add an appartement	✓ ↩
GET	/appartement/{appart_Id}	Get an appartement	✓ ↩
PUT	/appartement/{appart_Id}	Update an appartement	✓ ↩
DELETE	/appartement/{appart_Id}	Delete an appartement	✓ ↩

## Ticket Management

GET	/appartement/{appart_Id}/ticket	Get all tickets of an appartement	✓ ↩
POST	/appartement/{appart_Id}/ticket	Add a ticket to an appartement	✓ ↩
GET	/ticket/{ticket_Id}	Get a ticket of an appartement	✓ ↩
PUT	/ticket/{ticket_Id}	Update a ticket of an appartement	✓ ↩



## Repair estimate Management

GET	/ticket/{ticket_Id} /repair_estimate	Get all estimates repair of a ticket	✓ ↩
POST	/ticket/{ticket_Id} /repair_estimate	Add an estimate repair to a ticket	✓ ↩
DELETE	/repair_estimate/{estimate_Id}	Delete a estimate repair of a ticket	✓ ↩

## Renter Management

GET	/appartement/{appart_Id}/renter	Get the renter of an appartement	✓ ↩
POST	/appartement/{appart_Id}/renter	Add a renter to an appartement	✓ ↩
PUT	/appartement/{appart_Id}/renter	Update the renter of an appartement	✓ ↩
DELETE	/appartement/{appart_Id}/renter	Delete the renter of an appartement	✓ ↩

### Authentification :

L'authentification n'ayant pas été réfléchi lors de la phase de création du projet, celle-ci a été rapidement implémenté pour être en phase avec les exercices du cours plutôt qu'une réelle authentification. Ceci implique que dans le projet l'authentification est réalisée par une simple requête GET sur le chemin « /login », dans le monde réel il s'agit d'une faille de sécurité majeur.

Dans un monde idéal nous devrions créer dans la base de données une table users comportant une clé primaire, le nom d'utilisateur, et le Hash du mot de passe.

Lorsqu'un utilisateur souhaite s'identifier, le backEnd reçoit le nom d'utilisateur et le Hash du mot de passe qui sont placés dans les paramètres de la requête. Du côté backend il est recherché une correspondance entre le couple nom d'utilisateur et hash du mot de passe.

Lorsque la correspondance est effectuée le token d'identification est alors renvoyé, Sinon une erreur est envoyée pour un problème de correspondance.

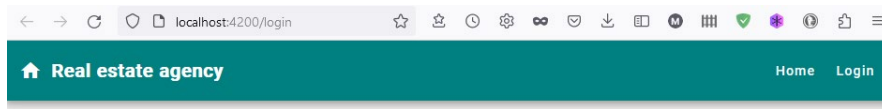
### FrontEnd Angular:

Github : [https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/tree/main/WEB/FrontEnd/real\\_estate\\_angular](https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/tree/main/WEB/FrontEnd/real_estate_angular)

Les chemins possibles sont les suivants :

*Login fig1 -> Home (Property List) fig2 -> Ticket fig3 -> Estimate fig4*  
*Home (Property List) fig2 -> Renter fig5*

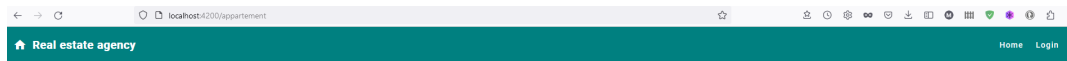
Il est également possible de rechercher directement un Ticket par son ID dans la page principale.



## Real estate login

**Login**

Figure 1: Page de login



## Property list

**Search a ticket of an appartement**

**Create a new property**

**Penthouse**  
Penthouse in the european district  
Price : 2000 €  
Surface : 100 m²  
Number of rooms : 4  
Address : Rue du blabla

**House**  
Clos house  
Price : 2500 €  
Surface : 150 m²  
Number of rooms : 6  
Address : Place du Blabla

Figure 2: Home, liste des biens

## Ticket list for the appartement ID : 1

Create a new ticket for Appartement ID : 1

Title

Description

Status

Create

Chauffage

Ticket ID : 2

Description : le chauffage ne fct pas

Status : Done

Appartement ID : 1

Edit

Estimates

Fenetre cassé

Ticket ID : 3

Description : la fenetre de la salle de bain est cassé

Status : new

Appartement ID : 1

Edit

Estimates

Figure 3: Liste des tickets

## Estimate list for the ticket ID : 2

Create a new estimate for Ticket ID : 2

Title

Description

Price

Create

chou

Branchement

Price : 300

Ticket ID : 2

Delete

Figure 4: Liste des devis

## Renter list for the appartement ID : 1

Create a new renter for Appartement ID : 1

First name

Last name

Email

Create

Tomate La sauce

Renter ID : 2

Email : lasauce.tomate@blabla.com

Appartement ID : 1

Delete

Update

Figure 5: Information locataire

## Architecture:

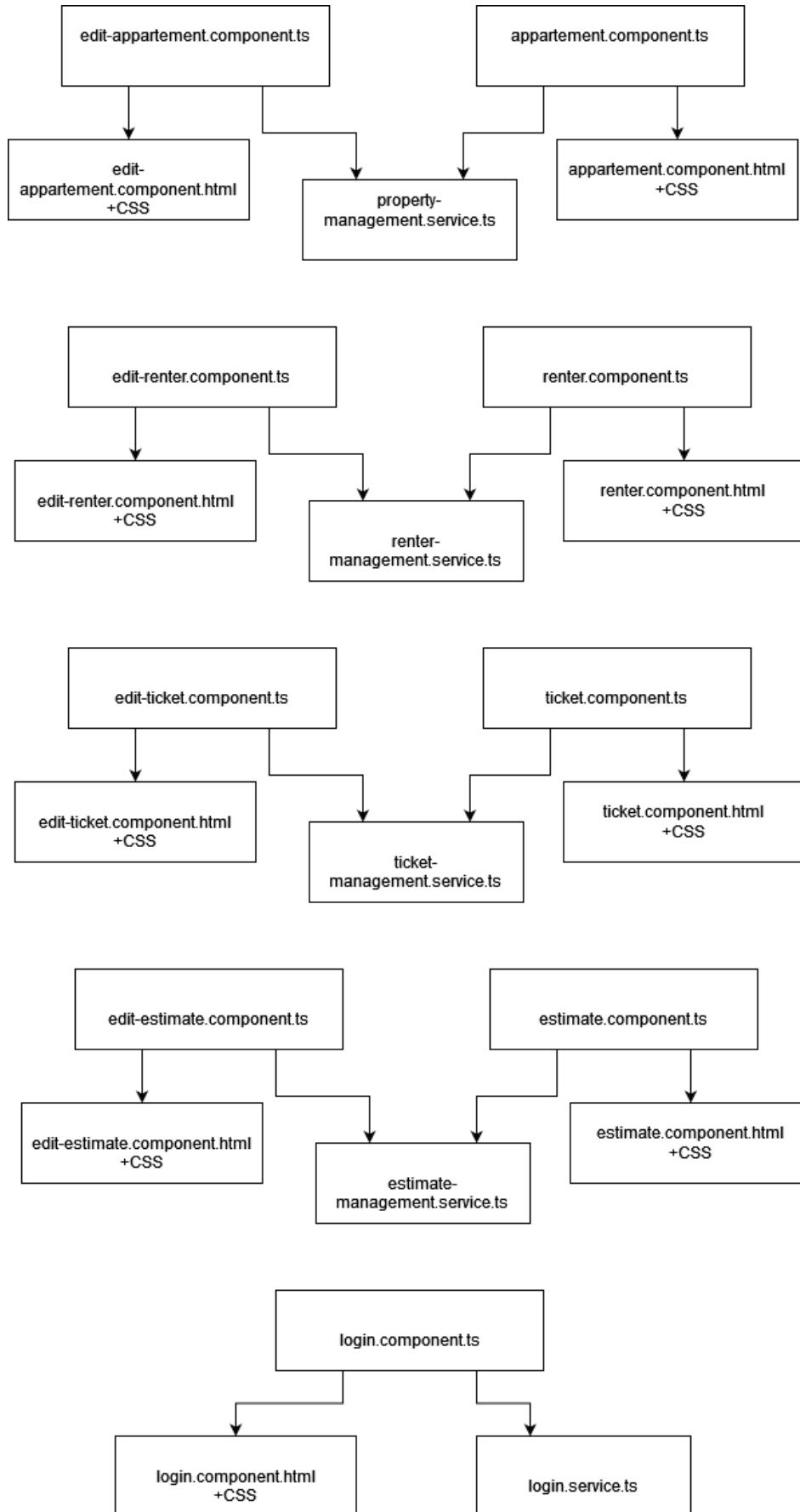


Figure 6: Architecture component + service

## FrontEnd App mobile hybride Ionic:

Github: [https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/tree/main/WEB/FrontEnd/real\\_estate\\_ionic](https://github.com/LeTouristeDeLECAM/Projet-Web-architecture-Agence-Immobiliere/tree/main/WEB/FrontEnd/real_estate_ionic)

La partie Ionic comporte des plusieurs problèmes, ceux-ci sont uniquement sur la page appartement :

- La fenêtre est divisée en trois parties
- Les liens de ne fonctionnement pas

Comme pour la partie WEB conventionnelle les chemins sont les suivants :

*Home (Property List) fig6 -> Ticket fig9 -> Estimate fig10*

*Home (Property List) fig6 -> Renter fig7*

*Login fig8*

The screenshot displays the 'Real Estate' app interface. On the left is a sidebar menu with the title 'Real Estate' and two items: 'Appartement' (with a house icon) and 'Login' (with a key icon). The main content area on the right is divided into three sections:

- Search a ticket of an apartment:** Contains a 'Ticket Id' input field and a 'SEARCH' button.
- Create a new property:** Contains several input fields for 'Title', 'Description', 'Price', 'Surface', 'Number ...', and 'Address', each with a corresponding label and a dropdown arrow. A 'CREATE' button is at the bottom.
- Penthouse:** A listing titled 'Penthouse in the european district' with details: 'Price : 2000 €', 'Surface : 100 m²', 'Number of rooms : 4', and 'Address : Rue du blabla'.

Figure 7: Home, liste des biens

Real Estate

Appartement

Login

Create a new renter for Appartement ID: 1

First nameFirst name

Last nameLast name

EmailEmail

CREATE

Tomate La sauce

Renter ID: 2

Email: lasauce.tomate@ciabia.com

Appartement ID: 1

DELETE

UPDATE

Figure 8: Information locataire

Real Estate

Appartement

Login

Login

User Na...User Name

PasswordPassword

LOGIN

Figure 9: Page de Login

Real Estate

Appartement

Login

Create a new ticket for Appartement ID: 1

TitleTitle

Descripti...Description

StatusStatus

CREATE

Chauffage

Ticket ID: 2

Description: le chauffage ne lct pas

Status: Done

Appartement ID: 1

EDIT

ESTIMATES

Fenetre cassé

Ticket ID: 3

Description: la fenetre de la salle de bain est cassé

Status: new

Appartement ID: 1

EDIT

ESTIMATES

Figure 10: Liste des tickets

Real Estate

Appartement

Login

### Estimate list for the ticket ID: 2

Create a new estimate for Ticket ID: 2

Title Title

Description Description

Price Price

CREATE

chou Branchement

Price: 300  
Ticket ID: 2

DELETE

Figure 11: Liste des devis

## Conclusion :

User stories	Priorité	API doc et dev	Frontend Angular	Reste à faire/ changement
<b>Gestion des classes de biens</b>				
En tant qu'agence immobilière, je veux pouvoir ajouter un bien d'un propriétaire dans un lot	1	0%	0%	Ne fait plus partie du projet
En tant qu'agence immobilière, je veux pouvoir supprimer un bien d'un lot	1	0%	0%	Ne fait plus partie du projet
<b>Gestion d'un bien</b>				
En tant qu'agence immobilière, je veux pouvoir crée un bien	1	100%	100%	
En tant qu'agence immobilière, je veux pouvoir supprimer un bien	1	100%	100%	
En tant qu'agence immobilière, je veux pouvoir modifier un bien	3	100%	90%	Afficher les anciennes données
<b>Gérer les locataires</b>				
En tant qu'agence immobilière, je veux pouvoir ajouter un locataire à un bien	4	100%	100%	
En tant qu'agence immobilière, je veux pouvoir supprimer un locataire d'un bien	4	100%	100%	
En tant qu'agence, je veux pouvoir modifier un locataire		100%	90%	Ajout au projet , afficher les anciennes données
<b>Gérer les problèmes technique</b>				

En tant qu'agence immobilière, je veux pouvoir créer un ticket pour un problème technique	2	100%	100%	
En tant qu'agence immobilière, je veux pouvoir supprimer un ticket	3	0%	0%	Ne fait plus partie du projet
En tant qu'agence immobilière, je veux pouvoir lister les tickets	2	100%	100%	
En tant qu'agence immobilière, je veux pouvoir associer un devis à un ticket	2	100%	100%	
<b>Gérer les devis</b>				
En tant qu'agence je veux pouvoir supprimer un devis		100%	100%	Ajout au projet
En tant qu'agence je veux pouvoir lister les devis		100%	100%	Ajout au projet
<b>Authentification</b>				
En tant qu'agence je veux pouvoir m'authentifier pour réaliser des modifications à la base de données		25%	90%	Côté Backend création d'une table User, Hash et recherche du match user et hash
<b>Rechercher</b>				
En tant qu'agence je veux pouvoir chercher un ticket par son ID		100%	100%	Ajout au projet

A mesure de l'avancement du projet beaucoup de nouveaux éléments ont été ajoutés et implémentés au projet, il en est de même pour les éléments supprimés du projet, ceci est dû à une meilleure compréhension du projet au fur et à mesure de son développement.

Pour exemple :

- Du côté backend il a été ajouté la modification des données d'un locataire, la suppression de devis et d'autres éléments. A contrario la création de lot et la modification des lots ont été supprimés du projet.
- Du côté Frontend il a été implémenté une authentification via token envoyée par le backend. Il reste encore à construire les tables et la recherche de correspondance entre user et hash du côté backend.

Ce dynamisme a été rendu possible par la démarche agile suivie tout au long du projet.

La symbiose entre la méthode agile et mon propre apprentissage ont permis de rediriger le projet pour qu'il corresponde aux exercices, avec du recul et des compétences accrues sur le sujet il est aujourd'hui plus facile d'imaginer créer un projet avec des bases solides en minimisant les changements et identifier les points prioritaires.