****

**Une image contenant texte, clipart

Description générée automatiquement**

**Rapport du projet tutoré**

**Spécialité Informatique**

****

**-- ImmoMarket --**

**Réalisés par :**

* Anas Ait Elhossaini
* Asaad Elkhal

**Encadré par :**

|  |  |  |
| --- | --- | --- |
| Pr. CHEFIRA Reda | Enseignant chercheur | Université Privée de Marrakech (UPM) |

# Remerciements:

Je tiens à exprimer ma profonde gratitude à **Pr. CHEFIRA Reda** pour son accompagnement, ses précieux conseils et son engagement tout au long de ce projet. Je remercie également l'**Université Privée de Marrakech (UPM)**, qui m’a offert un cadre propice à l’apprentissage et au développement de mes compétences. Enfin, je souhaite adresser mes sincères remerciements à mes collègues pour leur soutien, leur collaboration et les échanges enrichissants qui ont contribué à la réussite de cette expérience.

# Résumé:

Le but de cette application est de créer une marketplace immobilière intuitive et efficace qui connecte les clients, les entrepreneurs et l’administrateur. Grâce à une interface frontend développée en Vue.js et un backend en Flask, l'application permet aux clients de découvrir, louer ou acheter des biens immobiliers tels que des appartements et des maisons. Les entrepreneurs peuvent, de leur côté, créer et gérer des annonces pour les biens qu’ils souhaitent proposer. Enfin, l’administrateur supervise le fonctionnement global du site, notamment en gérant les utilisateurs, les annonces et le contenu, afin de garantir une expérience fluide et sécurisée pour tous les acteurs.

# Abstract:

This app aims to create an intuitive and efficient real estate marketplace that connects clients, entrepreneurs, and administrators. Leveraging a Vue.js-powered frontend and a Flask-based backend, the platform allows clients to explore, rent, or purchase properties such as apartments and houses. Entrepreneurs can list and manage properties they wish to offer, while the administrator oversees the platform's overall functionality, including user management, listings moderation, and content control. Additionally, the app incorporates responsive design principles, ensuring accessibility across various devices, including desktops, tablets, and smartphones. Advanced search and filtering features allow clients to easily find properties that meet their specific needs, while real-time notifications keep users updated on their activities and interactions. The platform also prioritizes data security and scalability, making it suitable for both small-scale and large-scale real estate markets. This comprehensive solution is designed to modernize the property transaction process, making it more transparent, user-friendly, and efficient for all stakeholders.

# Tables des matières:

[Remerciements: 2](#_Toc28466)

[Résumé: 3](#_Toc3768)

[Abstract: 4](#_Toc30907)

[Tables des matières: 5](#_Toc24607)

[Etude de l’existant 5](#_Toc31770)

[Diagrammes d’UML: 12](#_Toc28352)

[Implémentation: 21](#_Toc20912)

[ Sécurité et Authentification JWT: 25](#_Toc4500)

[ Analyse et Conception : Vue.js avec Node.js: 26](#_Toc30826)

[Tables des tableaux: 28](#_Toc13591)

[Liste des abréviations: 29](#_Toc31485)

[Sommaire: 30](#_Toc13616)

[Introduction générale: 31](#_Toc30104)

[Plan du rapport: 32](#_Toc28491)

[Chapitre 1: 34](#_Toc31132)

[Conclusion générale: 35](#_Toc15319)

[Annexes: 36](#_Toc15179)

[Bibliographie: 37](#_Toc28021)

# Etude de l’existant

### **Zillow:**

#### Points Forts :

**Recherche Avancée** : Zillow propose une recherche détaillée avec des filtres variés (prix, localisation, type de propriété, etc.) et une carte interactive pour visualiser les biens.

**Estimation de Prix (Zestimate)** : Zillow offre une estimation automatique de la valeur des propriétés, ce qui est utile pour les acheteurs et les vendeurs.

**Interface Utilisateur** : Le site est visuellement attrayant et facile à naviguer, avec une expérience utilisateur intuitive.

**Contenu Riche** : Les annonces sont généralement détaillées, avec de nombreuses photos, descriptions, et parfois des visites virtuelles.

**Outils pour les Acheteurs** : Zillow propose des calculateurs de prêts hypothécaires et des outils pour estimer les mensualités.

#### Points Faibles :

**Précision des Zestimates** : Les estimations de prix peuvent parfois être inexactes, ce qui peut causer de la confusion.

**Annonces Obsolètes** : Certaines annonces ne sont pas mises à jour régulièrement, ce qui peut frustrer les utilisateurs.

**Publicité Intrusive** : Le site contient beaucoup de publicités et de promotions, ce qui peut distraire l'utilisateur.

**Support Client** : Le support client n'est pas toujours réactif, surtout pour les problèmes techniques.

### **Realtor.com:**

#### Points Forts :

**Données Fiables** : Realtor.com est connu pour ses données immobilières précises et à jour, car il est directement lié aux bases de données des MLS (Multiple Listing Services).

**Recherche Géolocalisée** : Le site propose une excellente recherche géolocalisée, permettant aux utilisateurs de trouver des biens dans des zones spécifiques.

**Design Responsive** : Le site est bien optimisé pour les appareils mobiles, offrant une expérience utilisateur fluide sur tous les supports.

**Contenu Éducatif** : Realtor.com propose des guides et des articles utiles pour les acheteurs et les vendeurs, ce qui en fait une ressource éducative.

**Annonces Actualisées** : Les annonces sont généralement mises à jour en temps réel, ce qui réduit le risque de tomber sur des biens déjà vendus.

#### Points Faibles :

**Interface Moins Moderne** : Comparé à Zillow, l'interface de Realtor.com peut paraître un peu moins moderne et visuellement attrayante.

**Fonctionnalités Limitées** : Le site propose moins d'outils interactifs (comme les estimations de prix automatiques) que Zillow.

**Publicité** : Comme Zillow, Realtor.com contient des publicités qui peuvent parfois gêner la navigation.

**Options de Connexion** : Les options d'inscription et de connexion sont moins variées (principalement via email).

### Comparaison Générale :

**Zillow:** est plus axé sur l'expérience utilisateur et les outils interactifs, mais peut souffrir de problèmes de précision des données.

**Realtor.com:** est plus fiable en termes de données et de mises à jour, mais son interface et ses fonctionnalités sont moins innovantes.

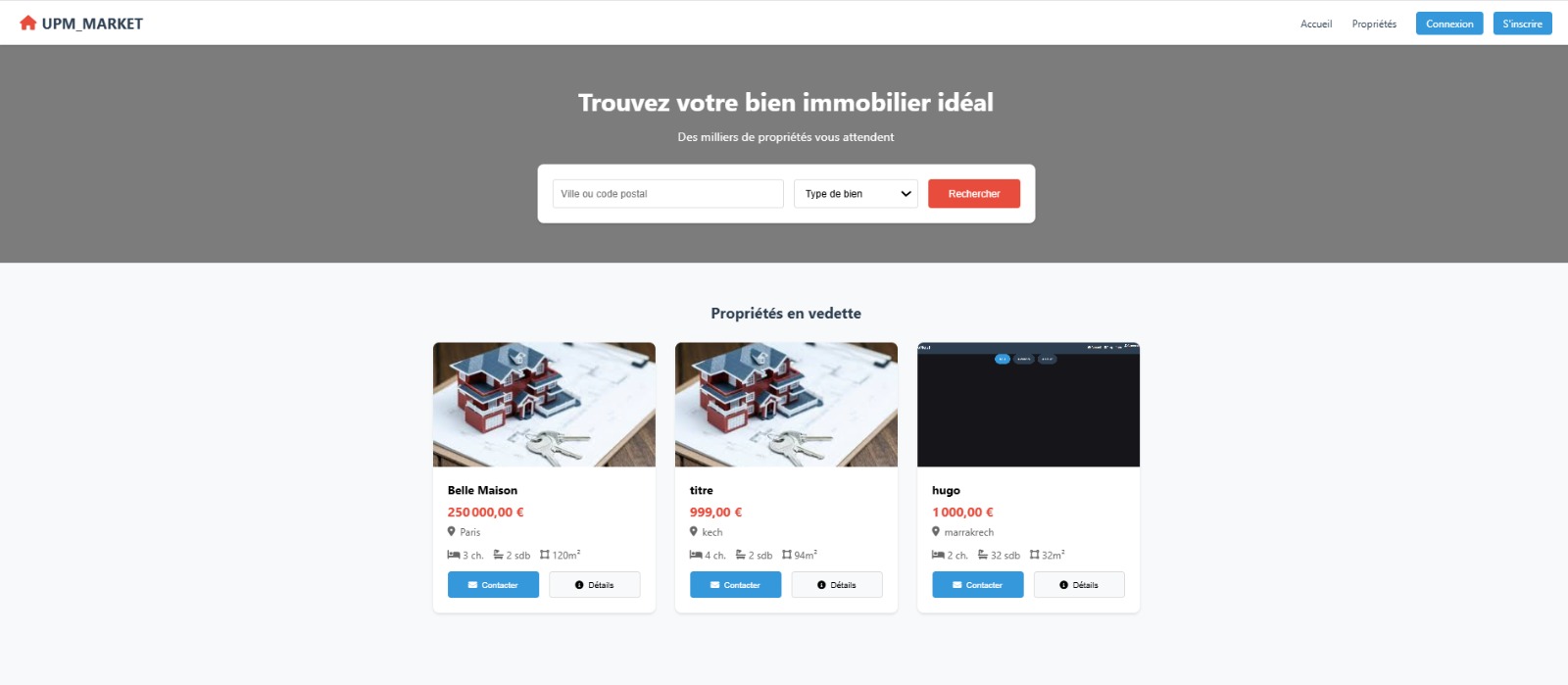
Liste des Figures:   
  


Figure 1:Home page

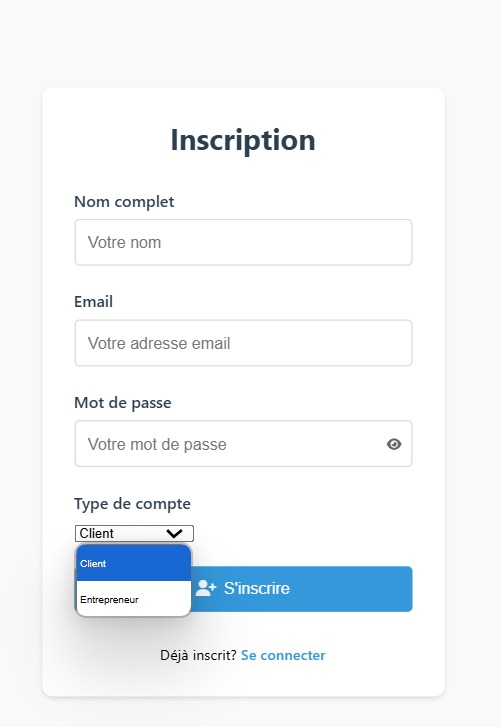


Figure 2:Interface d’inscription

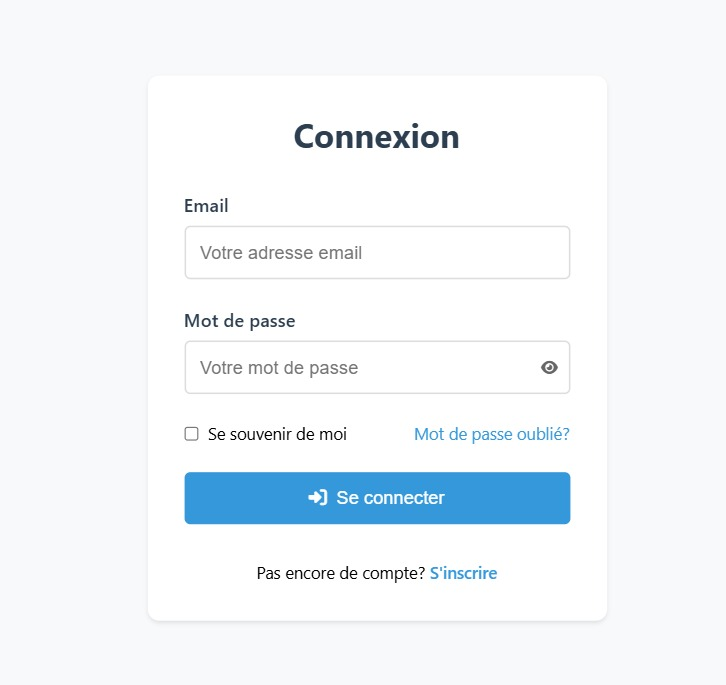


Figure 3:Interface d’authentification

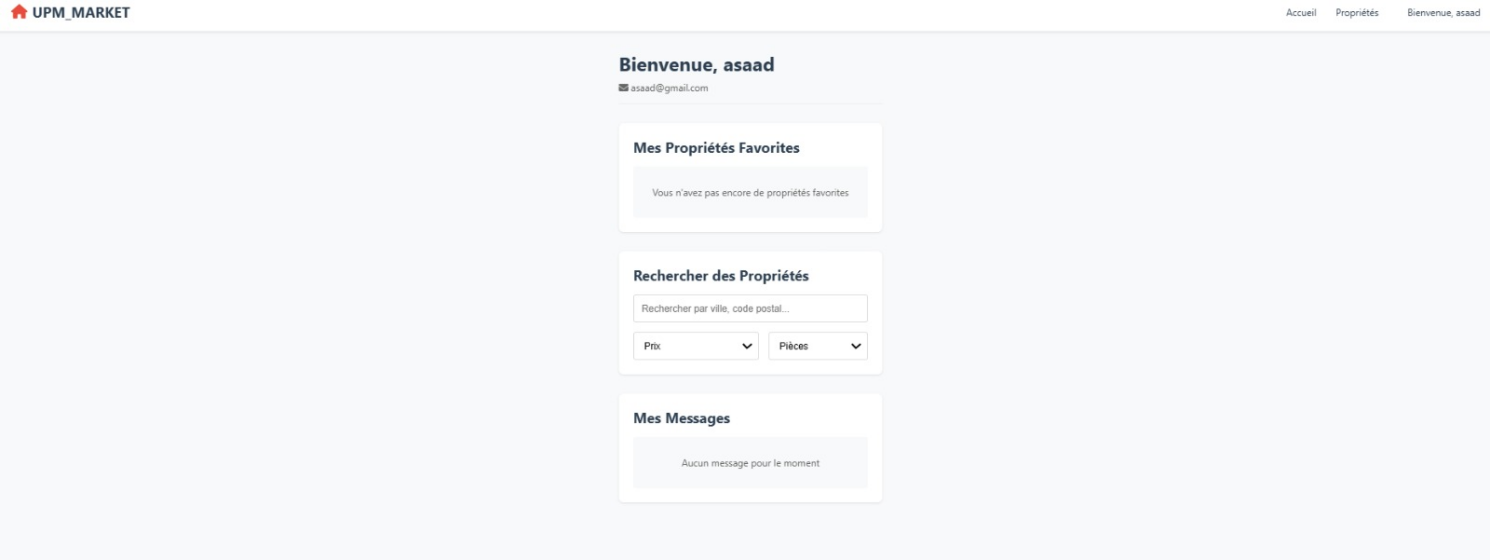


Figure 4:Interface client

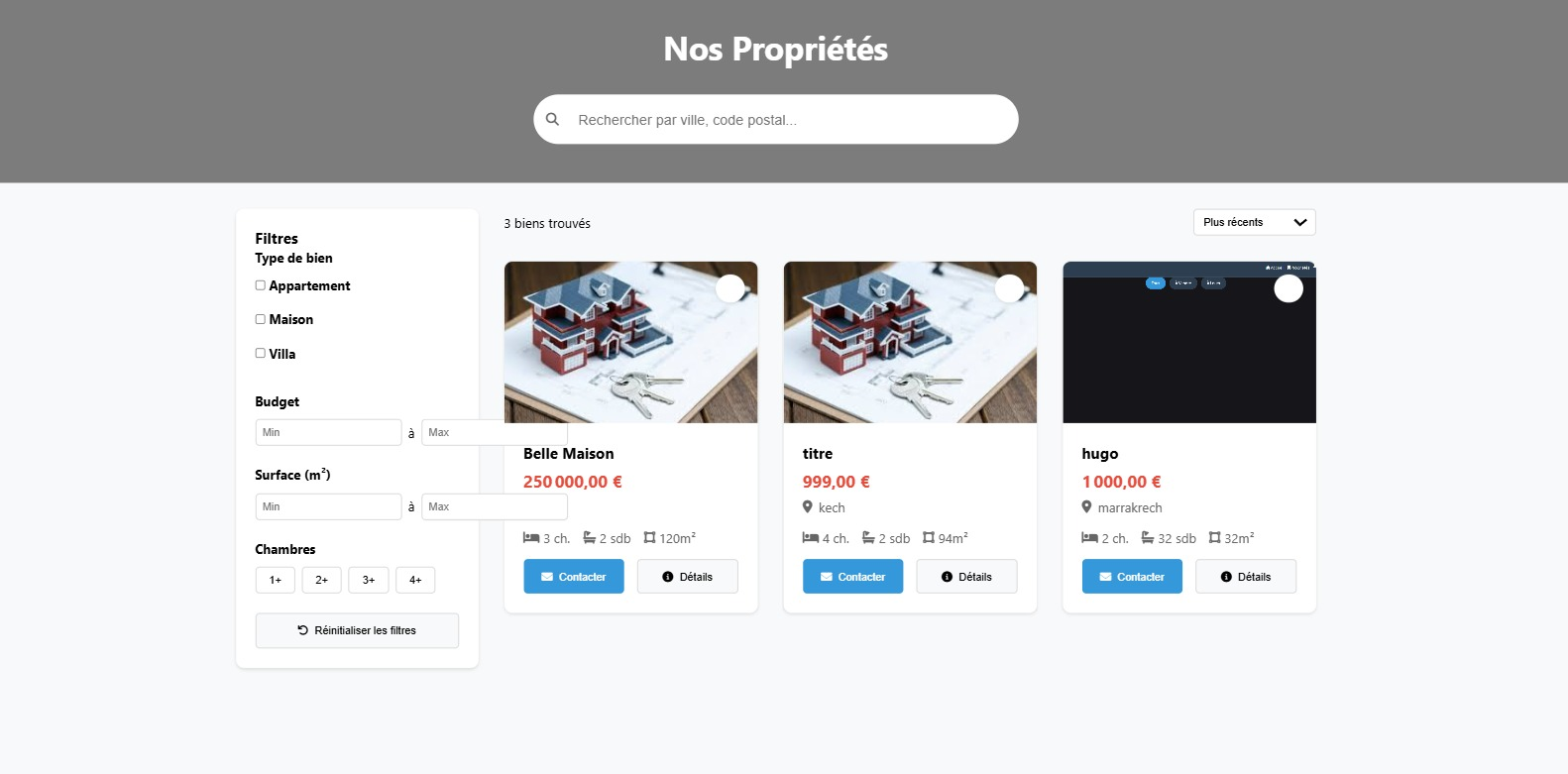


Figure 5:Interface des propriétés

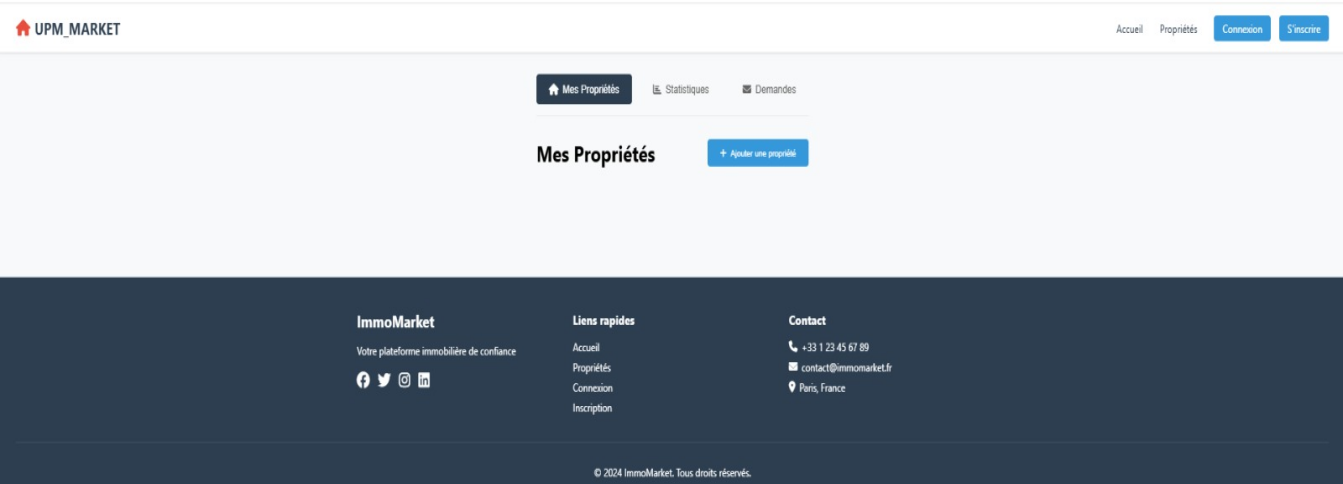


Figure 6: Dashboard entrepreneur

# Diagrammes d’UML:

Diagramme decas d’utilisation:

Ce **diagramme de cas d'utilisation** qui illustre les interactions entre différents types d'acteurs et les fonctionnalités d'un système. Voici une description détaillée des éléments présents dans l'image :

### **1. Gestion des Transactions:**

* **Acteur : Entrepreneur**
  + L'entrepreneur peut réaliser les actions suivantes :
    - **Suivre la Transaction** : Suivre l'état ou le progrès d'une transaction.
    - **Refuser une Offre** : Rejeter une offre reçue.
    - **Accepter une Offre** : Valider ou approuver une offre reçue.
    - **Faire une Offre** : Proposer une nouvelle offre à un autre utilisateur.

### **2. Gestion des Annonces:**

* **Acteur : Client**
  + Le client a accès aux fonctionnalités suivantes :
    - **Supprimer une Annonce** : Retirer une annonce existante.
    - **Modifier une Annonce** : Modifier le contenu ou les informations d'une annonce.
    - **Ajouter une Annonce** : Créer et publier une nouvelle annonce.
    - **Consulter les Annonces** : Parcourir ou afficher les annonces disponibles.
    - **Se Connecter** : Se connecter à son compte.
    - **Se Déconnecter** : Se déconnecter de son compte.

### **3. Gestion des Utilisateurs:**

* **Acteur : Admin**
  + L'administrateur est responsable des actions suivantes :
    - **Consulter le Profil** : Afficher les informations d'un profil utilisateur.
    - **Mettre à Jour le Profil** : Modifier ou mettre à jour les informations d'un profil utilisateur.
    - **S'inscrire** : Créer un compte utilisateur.
    - **Valider l'Inscription** : Approuver ou confirmer l'inscription d'un nouvel utilisateur.

### Observations générales :

1. Les **acteurs** sont représentés par des silhouettes (stick figures) : **Entrepreneur**, **Client**, et **Admin**.
2. Les groupes de cas d'utilisation sont divisés en trois systèmes principaux :
   * **Gestion des Transactions**
   * **Gestion des Annonces**
   * **Gestion des Utilisateurs**

😊

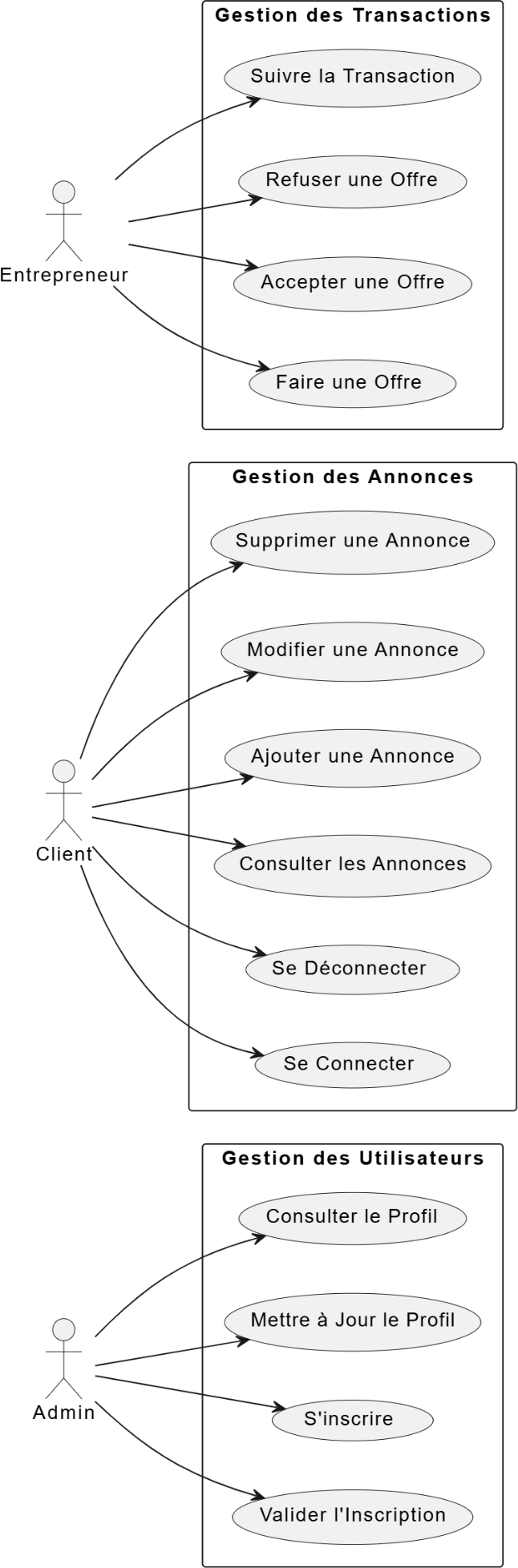


Figure 7

Diagramme de séquence:

Ce **diagramme de séquence UML** qui montre l'interaction entre les acteurs (Client, Entrepreneur, Admin) et les différentes fonctionnalités d'un système.

### **Acteurs :**

1. **Client** : Représenté par la première ligne de vie (colonne de gauche).
2. **Entrepreneur** : Représenté par la deuxième ligne de vie (colonne centrale).
3. **Admin** : Représenté par la troisième ligne de vie (colonne de droite).

Chaque acteur interagit avec le système selon des cas spécifiques.

### **Scénarios illustrés :**

#### **1. Inscription et Connexion**

* Le **Client** :
  + **S'inscrit** en envoyant une demande.
  + L'**Admin** valide l'inscription.
  + Le **Client** peut ensuite se **connecter** pour accéder au système.

#### **2. Gestion des Annonces**

**Client** :

* + Consulte les annonces disponibles (itération représentée par un bloc loop).
  + Fait une offre sur une annonce.
  + Si l'offre est acceptée (**alt** – alternative conditionnelle) :
    - Suit la transaction jusqu'à son aboutissement.
  + Si l'offre n'est pas acceptée :
    - Refait une offre.

**Entrepreneur** :

* + Peut ajouter, modifier ou supprimer des annonces.

#### **3. Gestion des Offres**

* **Client** :
  + Consulte les offres proposées dans un cycle (loop).
  + Peut accepter ou refuser une offre :
    - Si l'offre est **acceptée**, le processus suit cette branche conditionnelle.
    - Si l'offre est **refusée**, l'utilisateur en est informé.
  + Une fois les actions terminées, le **Client** peut se déconnecter.

### **Observations générales :**

1. Les blocs loop indiquent une répétition d'actions, par exemple, consulter plusieurs annonces ou offres.
2. Les blocs alt montrent des chemins conditionnels basés sur une décision (par exemple : offre acceptée ou refusée).
3. Les interactions entre les acteurs sont montrées sous forme de messages horizontaux (flèches).
4. Le diagramme est structuré en plusieurs segments pour représenter des fonctionnalités spécifiques :
   * Gestion des annonces.
   * Gestion des offres.
   * Inscription et validation.

Une image contenant texte, diagramme, capture d’écran, Parallèle

Description générée automatiquement

Figure 8

Diagramme d’activité:

Ce **diagramme d'activités UML**, qui décrit le flux des actions et des décisions dans un système, en mettant en évidence les interactions entre les différents utilisateurs et fonctionnalités. Voici une description détaillée :

### **Déroulement des activités :**

#### **1. Inscription et Validation**

* **Début** :
  + Le processus commence par l'action du **Client** qui s'inscrit.
* **Validation de l'inscription** :
  + L'**Admin** valide l'inscription.
  + Deux scénarios possibles :
    - Si l'inscription est validée : Le client peut passer à l'étape suivante.
    - Sinon : Le client reçoit un message d'erreur.

#### **2. Connexion et Navigation**

* Si l'inscription est validée, le **Client** peut se connecter.
* Une fois connecté, le client peut :
  + **Consulter les annonces**.
  + **Faire une offre** sur une annonce.

#### **3. Gestion des Offres**

* **Décision après une offre** :
  + Si l'offre est acceptée :
    - L'**Entrepreneur** suit la transaction.
  + Si l'offre n'est pas acceptée :
    - Le **Client** peut refaire une offre.
* À chaque étape, une question est posée : "Client souhaite continuer ?"
  + Si oui : Le flux reprend à une étape précédente.
  + Si non : Le client peut passer à d'autres actions.

#### **Gestion des Annonces par l'Entrepreneur**

* L'**Entrepreneur** peut effectuer les actions suivantes :
  + **Ajouter une annonce**.
  + **Modifier une annonce**.
  + **Supprimer une annonce**.

#### **5. Consultation des Offres**

* L'**Entrepreneur** peut également consulter les offres reçues.
* Décision après consultation :
  + Si l'offre est acceptée : Elle est validée.
  + Si l'offre est refusée : Elle est rejetée.
* À chaque étape, l'**Entrepreneur** peut choisir s'il souhaite continuer.

#### **6. Déconnexion**

* Une fois toutes les actions terminées, le **Client** peut se déconnecter.
* **Fin** : Le processus se termine.

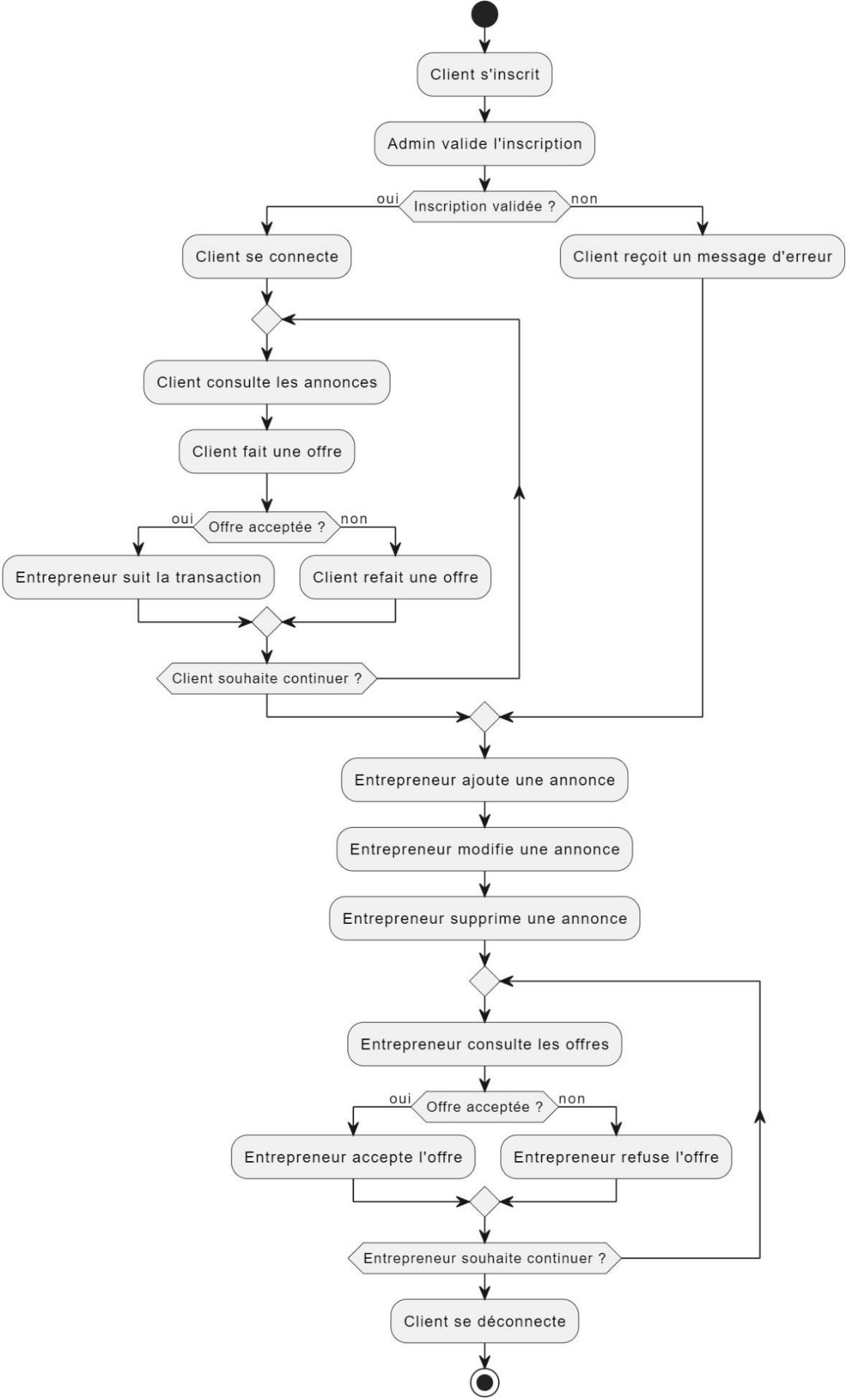


Figure 9

Implémentation:

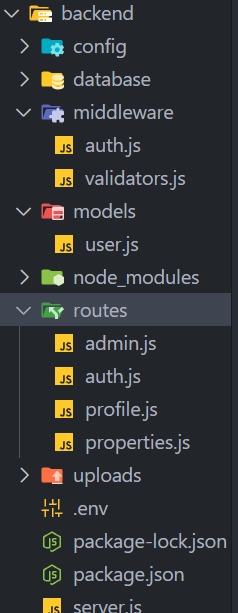


Figure 10: Dossier backend

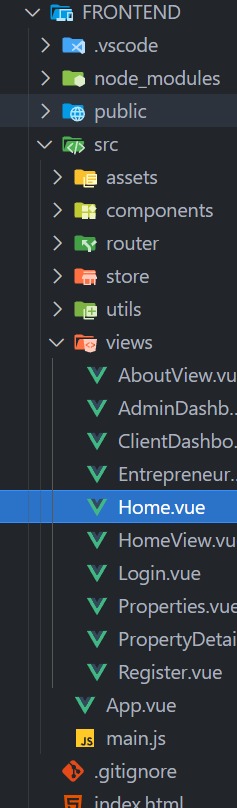


Figure 11:Dossier frontend

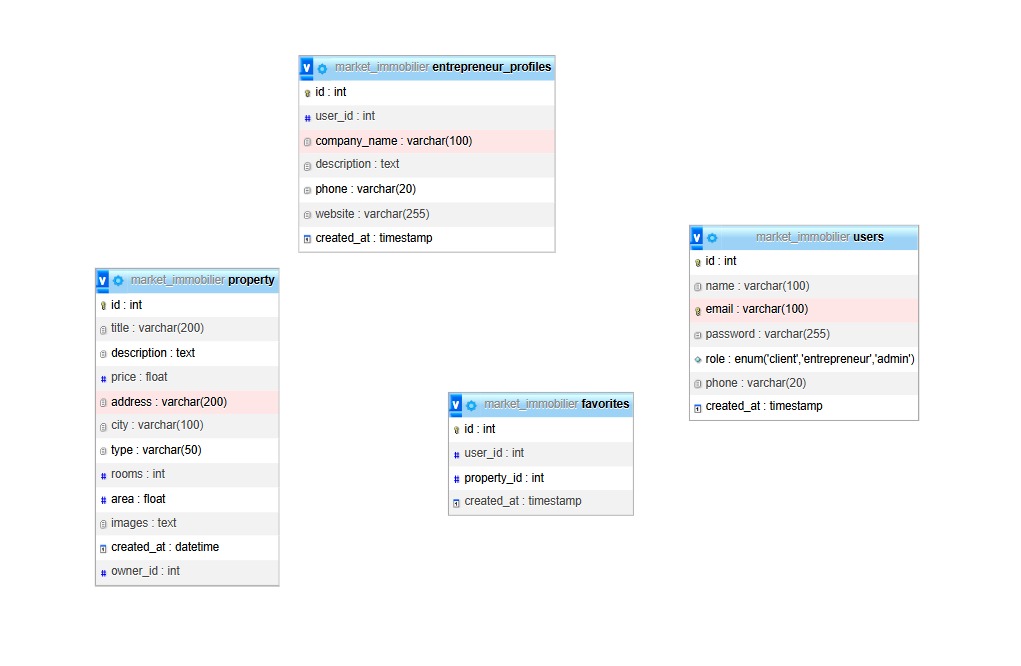


Figure 12: Base de données

Structure de la Base de Données ImmoMarket

1. Vue d'ensemble:

La base de données ImmoMarket est conçue pour gérer une plateforme immobilière avec trois types d'utilisateurs : clients, entrepreneurs et administrateurs. Elle comprend 4 tables principales interconnectées

2. Avantages de la Structure:

Modularité : Séparation claire des différentes entités

Extensibilité : Facilité d'ajout de nouvelles fonctionnalités

Sécurité : Gestion des rôles et permissions

Performance : Index optimisés pour les requêtes fréquentes

Maintenance : Structure claire et bien documentée

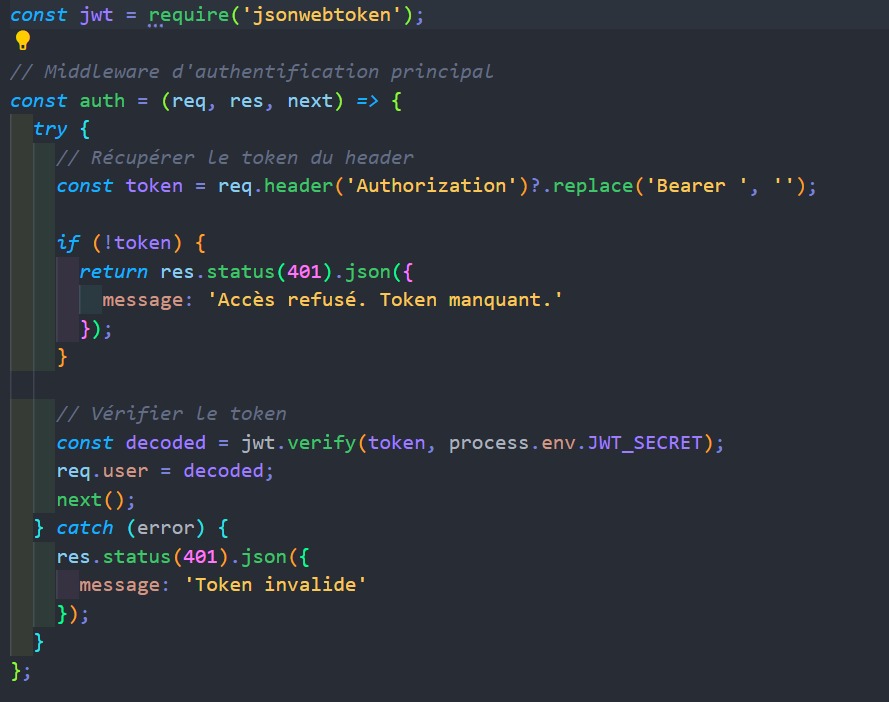
Cette structure permet une gestion efficace des :

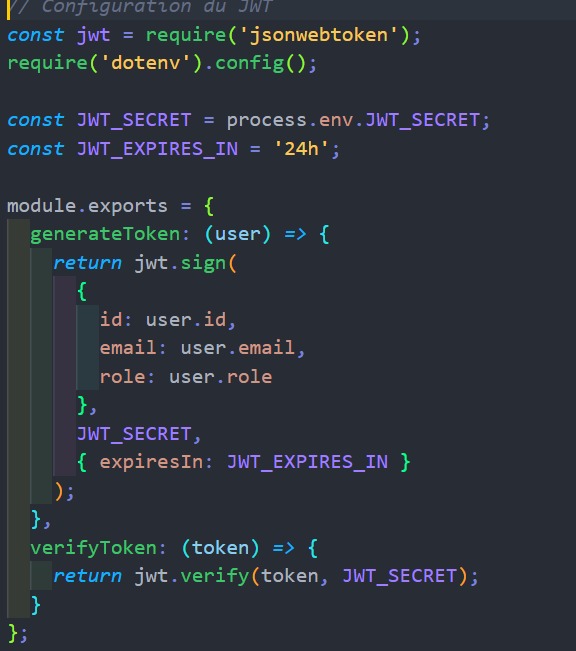
* Annonces immobilières
* Profils utilisateurs
* Relations client-entrepreneur
* Système de favoris
* Administration du site
* Sécurité et Authentification JWT:

Notre application implémente un système de sécurité robuste basé sur les JSON Web Tokens (JWT). Cette solution moderne d'authentification assure une protection complète des données et des accès utilisateurs. Le processus commence lors de la connexion où les credentials de l'utilisateur sont vérifiés. Une fois validés, un token JWT est généré, contenant les informations essentielles de l'utilisateur (ID, email, rôle) et une signature cryptographique unique. Ce token est ensuite utilisé pour sécuriser toutes les requêtes ultérieures via un middleware d'authentification qui vérifie systématiquement la validité du token et les permissions de l'utilisateur.

La sécurité est renforcée par une gestion fine des rôles (client, entrepreneur, administrateur) et des middlewares spécifiques qui contrôlent les accès aux différentes fonctionnalités. Par exemple, seuls les entrepreneurs peuvent gérer leurs propriétés, et seul l'administrateur peut accéder au dashboard d'administration. Les tokens ont une durée de validité limitée (24 heures) et sont vérifiés à chaque requête, assurant ainsi une protection continue contre les accès non autorisés.

Le système implémente également une gestion sécurisée des mots de passe avec hashage bcrypt, une protection contre les attaques CSRF, et des messages d'erreur sécurisés qui ne révèlent pas d'informations sensibles. Cette architecture de sécurité multicouche garantit la confidentialité des données et l'intégrité des communications client-serveur.





* Analyse et Conception : Vue.js avec Node.js:

Avantages :

1. Performance optimale :

Vue.js offre une interface utilisateur réactive, tandis que Node.js gère efficacement les requêtes côté serveur grâce à son architecture asynchrone.

2. Flexibilité et modularité:

Vue.js permet une structure basée sur des composants réutilisables, et Node.js offre une grande liberté pour construire des APIs personnalisées.

3. Écosystème JavaScript unifié :

Le développement full-stack JavaScript facilite la communication entre le frontend (Vue.js) et le backend (Node.js), simplifiant le partage de code et la maintenance.

4. Large communauté :

Les deux technologies bénéficient d'une communauté active et de nombreux outils, comme Vue CLI et Express.js.

5. SEO-friendly :

Avec \*Nuxt.js\* (framework basé sur Vue.js), vous pouvez améliorer le SEO grâce au rendu côté serveur (SSR), ce que Node.js facilite.

Inconvénients :

1. Complexité accrue pour les débutants :

La combinaison peut être difficile à maîtriser en raison des concepts avancés comme SSR, gestion d'état (Vuex), et le routage.

2. Dépendance à l'écosystème JavaScript :

Toute limitation ou problème lié à JavaScript affecte directement le projet.

3. Performance SSR :

Bien que puissant, le rendu côté serveur peut nécessiter une configuration complexe et impacter les performances si mal géré.

4. Gestion de la sécurité :

Node.js, en tant que backend, peut être vulnérable à des attaques courantes (injections SQL, XSS) si les bonnes pratiques ne sont pas respectées.

Conclusion :

Vue.js et Node.js forment une combinaison puissante et moderne pour le développement d'applications web réactives et performantes. Cependant, cette stack exige une bonne maîtrise de JavaScript et une gestion rigoureuse des performances et de la sécurité.

# Tables des tableaux:

Tableau 1 : Comparaison des technologies front-end.

Tableau 2 : Structure des tables de la base de données SQLite3.

Tableau 3 : Répartition des rôles et permissions.

Tableau 4 : Performances avant/après optimisation.

# Liste des abréviations:

# MVC : Model-View-Controller

# JWT : JSON Web Token

# CRUD : Create, Read, Update, Delete

# UI/UX : User Interface/User Experience

# API : Application Programming Interface

# CORS : Cross-Origin Resource Sharing

# Sommaire:

Introduction générale :

- Contexte du projet

- Objectifs pédagogiques

- Présentation succincte du projet

Description des missions et des tâches réalisées :

- Objectifs et contexte des tâches

- Détail des tâches effectuées

- Technologies utilisées (Vue.js, Flask, base de données)

Projets spécifiques : Marketplace immobilière :

- Description détaillée du projet

- Fonctionnalités principales (interface client, entrepreneur, administrateur)

- Rôles et contributions de chaque membre de l'équipe

- Difficultés rencontrées et solutions apportées

Analyse des compétences et connaissances acquises :

- Compétences techniques développées (Vue.js, Flask, gestion de base de données)

- Compétences managériales et personnelles (travail d'équipe, gestion du temps)

- Comparaison avec les objectifs initiaux du projet

Conclusion générale :

- Synthèse de l'expérience

- Apport du projet pour le projet professionnel futur

- Suggestions pour les futurs étudiants

Annexes :

- Captures d'écran de l'application

- Extraits de code (exemples de code Vue.js, Flask, SQL)

- Tableaux ou graphiques (si disponibles)

Bibliographie :

- Références aux documents et ouvrages consultés

# Introduction générale:

Dans un monde où la digitalisation transforme chaque aspect de nos vies, le marché immobilier ne fait pas exception. Cette application de marketplace immobilière vise à moderniser et simplifier les interactions entre les différentes parties prenantes dans le domaine de l'immobilier. Que ce soit pour louer ou acheter un bien, les clients peuvent explorer facilement une large gamme d'offres. Les entrepreneurs, quant à eux, ont une plateforme intuitive pour promouvoir leurs biens. Grâce à une gestion centralisée par l'administrateur, l'application garantit une expérience utilisateur fiable et fluide, tout en favorisant des transactions sécurisées. Le choix de Vue.js pour l'interface utilisateur et Flask pour la logique métier offre un environnement robuste et flexible pour cette application innovante.

# Plan du rapport:

## Introduction générale

## Description des missions et des tâches réalisées

## UI/UX

## Analyse des compétences et connaissances acquises

## Conclusion générale

## Annexes

## Bibliographie

🡪 Il est important de structurer le document de manière claire et précise. Voici un plan type que vous pourriez utiliser pour votre rapport de stage :

**1. Page de couverture**

* Titre du rapport
* Nom et logo de l’application
* Noms des étudiants et leurs coordonnées
* Période du projet
* Nom de l'école et année d'étude

**2. Remerciements**

* Expression de gratitude envers ceux qui ont contribué à la réussite du projet.

**3. Résumé**

* Un bref résumé du contenu du rapport, incluant les principales tâches réalisées et les compétences acquises.

**4. Table des matières**

* Liste de tous les titres de sections avec les numéros de page correspondants.

**5. Introduction**

* Contexte du projet (objectifs pédagogiques, pourquoi cette thématique).
* Présentation succincte du projet

**6. Description des missions et des tâches réalisées**

* Détail des tâches confiées, avec objectifs et contextes.
* Description des tâches effectuées jour après jour ou semaine après semaine.
* Technologies utilisées et compétences développées.

**7. Projets spécifiques**

* Description détaillée des projets sur lesquels vous avez travaillé.
* Votre rôle et contributions spécifiques dans ces projets.
* Difficultés rencontrées et solutions apportées.

**8. Analyse des compétences et connaissances acquises**

* Réflexion sur ce que le projet a apporté en termes de compétences techniques, managériales et personnelles.
* Comparaison avec les objectifs initiaux du projet.

**9. Conclusion**

* Synthèse de l'expérience.
* Apport du stage pour votre projet professionnel futur.
* Suggestions pour les futurs étudiants ou améliorations à faire.

**10. Annexes**

* Documents ou codes sources produits, graphiques, tableaux.
* Tout document supplémentaire qui appuie votre rapport.

**11. Bibliographie**

* Références à des documents ou ouvrages consultés durant le stage.

# Chapitre 1:

Ce chapitre est consacré à la présentation du projet ImmoMarket, une plateforme web innovante qui connecte clients, entrepreneurs, et administrateurs dans le domaine immobilier.

Le projet a pour objectifs :

1. Simplifier la recherche de propriétés.

2. Faciliter la gestion des annonces immobilières.

3. Offrir un système sécurisé et adapté aux différents rôles utilisateurs.

L’introduction aborde :

Le contexte général de l’immobilier digitalisé.

La problématique : manque de plateformes intégrées multi-rôles.

La solution proposée : Immomarket, alliant modularité, sécurité, et facilité d’utilisation.

# Conclusion générale:

Ce projet de marketplace immobilière a été une expérience enrichissante, tant sur le plan technique que personnel. Grâce à l'utilisation de technologies modernes comme **Vue.js** pour le frontend et **Node.Js** pour le backend, nous avons réussi à développer une plateforme fonctionnelle et intuitive, répondant aux besoins des clients, des entrepreneurs et des administrateurs. Les défis techniques, tels que l'intégration frontend/backend, la gestion des notifications en temps réel et la sécurisation des données, ont été surmontés grâce à une approche méthodique et une collaboration étroite.

Le **travail d'équipe** a joué un rôle central dans la réussite de ce projet. Chaque membre a apporté ses compétences et sa créativité, permettant de combiner nos forces pour atteindre les objectifs fixés. Cette collaboration a renforcé notre capacité à communiquer, à résoudre des problèmes complexes et à gérer efficacement notre temps.

En somme, ce projet a été une opportunité précieuse pour mettre en pratique nos connaissances, développer de nouvelles compétences et démontrer l'importance de la coopération dans la réalisation d'un objectif commun. Il marque une étape significative dans notre parcours académique et professionnel.

# Annexes:

Les annexes incluent :

Les diagrammes de l’architecture.

Les captures d’écran des principales interfaces.

Les extraits de code importants (e.g., configuration des routes, gestion de l’authentification).

La liste des dépendances utilisées.

# Bibliographie:

1. Documentation officielle Vue.js 3 : https://vuejs.org

2. Guide Node.js : https://nodejs.org

3. Tutoriels Express.js : https://expressjs.com

4. Références SQLite : https://sqlite.org

5. Articles sur l’authentification JWT : https://jwt.io