

**DIPLÔME UNIVERSITAIRE DE TECHNOLOGIE**

**FILIERE : GENIE INFORMATIQUE**

**RÉALISÉ AU SEIN:**

**ÉCOLE SUPÉRIEURE DE TECHNOLOGIE-SAFI**

# RAPPORT DE STAGE

**SUJET :**

**DÉVELOPPEMENT D'UN SITE WEB DE GESTION  
DES BIENS IMMOBILIERS.**

**ENCADRÉ PAR:**  
**JAMAL BAKKAS**

**RÉALISÉ PAR:**  
**AKRAM ELBAHAR**

# Remerciements:

Je tiens tout d'abord à exprimer ma profonde gratitude envers mon professeur et superviseur **Monsieur JAMMAL BAKKAS**, tout au long de ce stage, pour son soutien, ses conseils éclairés et sa disponibilité. Son encadrement a grandement contribué à l'aboutissement de ce projet et à l'enrichissement de mes compétences techniques et professionnelles.

Je remercie également École Supérieure De Technologie SAFI et l'ensemble de l'équipe pédagogique pour m'avoir offert l'opportunité de réaliser ce stage dans un environnement stimulant et propice à l'apprentissage. Leur confiance et leur soutien ont été essentiels au bon déroulement de ce travail.

# Résumé :

Ce rapport présente le travail effectué dans le cadre de mon stage au sein de ESTS sous la supervision de Monsieur JAMAL BAKKAS. L'objectif principal de ce stage était de concevoir et de développer un site web de gestion des biens immobiliers. Ce projet avait pour but de permettre aux utilisateurs de consulter et gérer des annonces immobilières, tout en facilitant les interactions entre les différentes parties prenantes, à savoir les utilisateurs, les agents immobiliers et les administrateurs.

Pour répondre aux exigences du cahier des charges, j'ai travaillé sur la mise en place d'une architecture web robuste, en utilisant les technologies JavaScript, Node.js, React.js et MongoDB. Le projet a impliqué le développement de fonctionnalités essentielles telles que la gestion des annonces, la messagerie interne, l'inscription des utilisateurs, ainsi que la modération des annonces par les administrateurs.

Ce stage m'a permis d'acquérir des compétences techniques en développement web, ainsi qu'une meilleure compréhension des processus de gestion de projet. Grâce à l'encadrement de mon professeur, j'ai pu surmonter les défis rencontrés et mener à bien ce projet qui pourra potentiellement être utilisé par l'école pour des projets futurs.

## Table des matières

<b>Introduction :</b> .....	7
<b>Contexte et objectif du stage:</b> .....	7
<b>Présentation de l'entreprise (l'école) :</b> .....	7
<b>L'École Supérieure de Technologie de Safi :</b> .....	7
<b>Description Générale du Projet:</b> .....	8
<b>Objectifs du projet :</b> .....	8
1. <b>Faciliter la publication et la gestion des annonces immobilières</b> .....	8
2. <b>Améliorer l'accès à l'information pour les utilisateurs</b> .....	8
3. <b>Optimiser la gestion administrative</b> .....	8
4. <b>Services offerts par la plateforme</b> .....	8
<b>Technologies utilisées (MERN):</b> .....	9
<b>BACKEND :</b> .....	9
<b>FRONTEND :</b> .....	9
<b>Base de données :</b> .....	10
<b>BACKEND :</b> .....	11
<b>Architecture Backend:</b> .....	11
Routes et leurs fonctions : .....	11
Routes d'authentification (/api/auth) : .....	11
Routes d'administration (/api/admin): .....	11
<b>Contrôleurs (Controllers) :</b> .....	13
<b>Exemple de Contrôleur des annonces (Ads Controller) :</b> .....	13
<b>Sécurité et Authentication:</b> .....	14
<b>Frontend:</b> .....	15
<b>Architecture Frontend:</b> .....	15
<b>Pages Principales du Site Web :</b> .....	15
<b>Home (Page d'accueil):</b> .....	15
<b>SignUp (Inscription):</b> .....	16
<b>Login (Connexion):</b> .....	16
<b>Dashboard des non administrateur (Tableau de Bord):</b> .....	17
<b>InsertAd (Insertion d'une annonce):</b> .....	18
<b>EditAd (Modifier une annonce):</b> .....	18

<b>AddDetails (Détails de l'annonce) :</b>	19
Affiche les détails complets d'une annonce, incluant le prix, la description, les photos, etc. ....	19
<b>Chat (Messagerie):</b>	19
<b>AdminDashboard (Tableau de bord Administrateur) :</b>	20
<b>EditProfile (Modification de profil):</b>	22
<b>LogOut (Déconnexion) :</b>	22
<b>Style et UI avec Tailwind CSS :</b>	22
<b>Base de Données:</b>	23
<b>Schéma des Annonces (Advertisement Schema) :</b>	23
<b>Schéma des Conversations (Conversation Schema) :</b>	24
<b>Schéma des Vues (Seen Schema) :</b>	24
<b>Schéma des Messages (Message Schema) :</b>	24
<b>Schéma des Utilisateurs (User Schema) :</b>	25
<b>Défis et Solutions:</b>	26
<b>Problèmes rencontrés:</b>	26
<b>Solutions apportées:</b>	26
<b>Conclusion:</b>	27
<b>Bilan du stage:</b>	27
<b>Acquis techniques et professionnels:</b>	27
Liste des figures :	6
Bibliographie:	28

## Liste des figures :

Figure 1 .....	9
Figure 1 .....	9
Figure 2 .....	9
Figure 2 .....	9
Figure 3 .....	9
Figure 3 .....	9
Figure 4 .....	9
Figure 4 .....	9
Figure 5 .....	10
Figure 5 .....	10
Figure 6 .....	15
Figure 7 .....	16
Figure 8 .....	16
Figure 9 .....	17
Figure 10 .....	17
Figure 11 .....	17
Figure 12 .....	18
Figure 13 .....	18
Figure 14 .....	19
Figure 15 .....	19
Figure 16 .....	20
Figure 17 .....	20
Figure 18 .....	21
Figure 19 .....	21
Figure 20 .....	22

# **Introduction :**

## **Contexte et objectif du stage:**

Dans le cadre de ma formation à l'École Supérieure des Sciences et Technologies (ESTS), j'ai eu l'opportunité d'effectuer un stage pratique d'une durée d'un mois au sein de l'établissement. Ce stage a été réalisé sous la supervision de Monsieur JAMAL BAKKAS, enseignant et responsable du projet.

L'objectif principal de ce stage était de développer un site web de gestion des biens immobiliers. Ce projet visait à appliquer les compétences techniques acquises durant ma formation dans un environnement réel, tout en répondant à un besoin concret identifié par l'école : la mise en place d'une plateforme facilitant la gestion et la communication autour des biens immobiliers.

Le projet m'a permis de travailler avec des technologies modernes telles que Node.js, Express.js, React, Tailwind CSS, et MongoDB, en relevant les défis liés au développement d'une site web complet et performant.

## **Présentation de l'entreprise (l'école) :**

### **L'École Supérieure de Technologie de Safi :**

Fondée en septembre 1992, l'École Supérieure de Technologie de Safi (ESTS) représente le premier noyau universitaire de la ville de Safi. Elle est rattachée à l'université Cadi Ayyad et se distingue comme un acteur clé dans la formation technologique et professionnelle de courte durée.

Depuis sa création, l'ESTS s'est fixée pour mission principale la formation de techniciens supérieurs hautement qualifiés à travers la préparation du Diplôme Universitaire de Technologie (DUT) dans diverses spécialités industrielles et tertiaires. À partir de l'année universitaire 2014-2015, et suite à une modification du décret régissant la vocation des établissements universitaires, l'ESTS a également introduit la formation en Licence Professionnelle (LP), enrichissant ainsi son offre de formation.

En plus de la formation initiale, l'ESTS a pour missions la recherche scientifique, la formation continue, la prestation de services, ainsi que la diffusion du savoir, de la culture et des connaissances. L'école veille à ce que le choix de ses filières s'aligne sur les besoins du développement économique du pays, garantissant ainsi une formation polyvalente. Cette approche permet aux lauréats

de l'ESTS de maximiser leurs chances d'insertion professionnelle tout en leur offrant une base solide pour la poursuite d'études supérieures.

## Description Générale du Projet:

### Objectifs du projet :

Le projet avait pour principal objectif de développer une plateforme web de gestion des biens immobiliers pour l'École Supérieure des Sciences et Technologies (ESTS). Cette plateforme vise à faciliter la gestion et la communication entre les différents acteurs impliqués dans la transaction immobilière.

Les objectifs spécifiques du projet sont :

#### 1. Faciliter la publication et la gestion des annonces immobilières :

- Offrir aux **agents immobiliers** un espace dédié pour publier, modifier et supprimer des annonces.
- Fournir des outils de gestion pour suivre les performances des annonces (nombre de vues, contacts).

#### 2. Améliorer l'accès à l'information pour les utilisateurs :

- Permettre aux **utilisateurs finaux** (acheteurs, locataires) de rechercher facilement des biens immobiliers en utilisant des filtres variés (type de bien, localisation, prix, etc.).
- Offrir la possibilité de sauvegarder des annonces en favoris et de contacter les agents directement via un système de messagerie intégré.

#### 3. Optimiser la gestion administrative :

- Mettre à disposition des **administrateurs** un tableau de bord complet pour gérer les utilisateurs et les annonces.
- Assurer la modération des contenus publiés sur la plateforme.

#### 4. Services offerts par la plateforme :

- **Recherche et Listing des Biens** : Une fonctionnalité de recherche avancée permettant aux utilisateurs de filtrer les annonces selon divers critères (surface, prix, nombre de pièces, etc.).
- **Publication et Gestion des Annonces** : Un espace pour les agents immobiliers afin de créer et gérer leurs annonces avec des options pour ajouter des descriptions détaillées, des photos, et des vidéos.



- **Messagerie Interne** : Un système de communication intégré permettant aux utilisateurs de contacter les agents et de gérer les interactions.
- **Tableau de Bord Administrateur** : Un espace pour les administrateurs leur permettant de superviser les activités du site, de gérer les utilisateurs, et de modérer les annonces.

En somme, le projet vise à créer une plateforme centralisée et efficace qui améliore la gestion des biens immobiliers, facilite la communication entre les utilisateurs, et optimise les processus administratifs.

## Technologies utilisées (MERN):

### BACKEND :



Figure 1

- *Node.js* : Utilisé comme environnement d'exécution pour développer le serveur backend. Il permet une gestion asynchrone des requêtes et offre une grande scalabilité.



Figure 3

- *Express.js* : Framework web minimaliste pour Node.js, utilisé pour créer des routes, gérer les requêtes HTTP et simplifier le développement des API REST.

### FRONTEND :

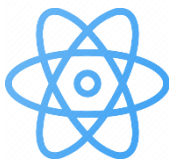


Figure 5

- *React* : Bibliothèque JavaScript utilisée pour créer des interfaces utilisateur dynamiques et réactives. Elle permet de développer des composants réutilisables et d'assurer une mise à jour rapide du DOM virtuel.



Figure 7

- *Tailwind CSS* : Un framework CSS utilitaire qui permet de concevoir des interfaces de manière rapide et efficace avec des classes pré-définies pour la mise en page et le style.

## Base de données :



Figure 9

- *MongoDB* : Base de données NoSQL utilisée pour stocker les données du site. Elle offre une grande flexibilité pour la gestion des documents et des collections, facilitant ainsi la scalabilité et l'adaptation aux données non structurées.

Figure 10

# BACKEND :

## Architecture Backend:

Le backend de ce projet a été conçu en utilisant **Node.js** et **Express.js** pour gérer les requêtes HTTP et fournir des données au frontend à travers une API RESTful. L'architecture suit le modèle MVC (Modèle-Vue-Contrôleur) qui permet de séparer la logique métier, les vues et l'accès aux données. Cela facilite la maintenance et l'évolution du projet.

Le backend est responsable de la gestion des utilisateurs, des annonces immobilières, des messages, ainsi que de l'authentification et des autorisations. Il communique avec une base de données **MongoDB** pour stocker et récupérer les informations.

## Routes et leurs fonctions :

Les routes sont utilisées pour définir les points d'accès à l'API, permettant aux utilisateurs et au frontend de communiquer avec le serveur. Voici une liste des principales routes et leurs fonctions :

### Routes d'authentification (/api/auth) :

- **POST /api/auth/signup** : Crée un nouveau compte utilisateur en enregistrant les informations (nom, email, mot de passe).
- **POST /api/auth/login** : Authentifie un utilisateur en vérifiant les informations d'identification (email, mot de passe) et renvoie un jeton JWT.
- **POST /api/auth/logout** : Déconnecte un utilisateur en détruisant la session active.

### Routes d'administration (/api/admin):

- **GET /api/admin/users** : Récupère la liste de tous les utilisateurs pour la gestion par les administrateurs.
- **GET /api/admin/user/** : Récupère les informations détaillées d'un utilisateur spécifique.
- **POST /api/admin/user** : Crée un nouveau compte utilisateur à des fins administratives.
- **PUT /api/admin/user/** : Modifie les informations d'un utilisateur spécifique.

- **DELETE /api/admin/user/**: Supprime un utilisateur en fonction de son ID.
- **GET /api/admin/advertisements** : Récupère la liste des annonces immobilières pour modération.
- **GET /api/admin/advertisement/** : Récupère les détails d'une annonce spécifique pour gestion.
- **DELETE /api/admin/advertisement/** : Supprime une annonce en fonction de son ID.
- **POST /api/admin/activateAd/** : Active ou désactive une annonce immobilière.
- **GET /api/admin/stats/** : Récupère des statistiques sur les utilisateurs, les annonces et les interactions selon une période donnée (aujourd'hui, derniers 7 jours, etc.).

#### **Routes des annonces (/api/advertisement) :**

- **GET /api/advertisement/** : Récupère les détails d'une annonce en fonction de son ID.
- **POST /api/advertisement** : Crée une nouvelle annonce immobilière.
- **PUT /api/advertisement** : Modifie une annonce existante.
- **DELETE /api/advertisement/** : Supprime une annonce spécifique.
- **GET /api/advertisement/favorite/** : Ajoute une annonce aux favoris d'un utilisateur.

#### **Routes des utilisateurs (/api/user) :**

- **GET /api/user** : Récupère les informations de profil d'un utilisateur connecté.
- **PUT /api/user** : Modifie les informations de profil d'un utilisateur.
- **GET /api/user/conversation** : Récupère la liste des conversations de l'utilisateur avec le dernier message.
- **GET /api/user/userfavseen** : Récupère les annonces que l'utilisateur a marquées comme vues ou favorites.

- **POST /api/user/searchUsername** : Recherche un utilisateur par nom d'utilisateur.

#### Routes de messagerie (/api/message) :

- **POST /api/message/send/** : Envoie un message à un autre utilisateur.
- **GET /api/message/** : Récupère la liste des messages échangés avec un utilisateur spécifique.

#### Routes de gestion des fichiers (/upload) :

```
export const getAds = async (req, res) => { try { const offset =
parseInt(req.query.offset) || 0; // Récupère l'offset pour la pagination const
ads = await Advertisement.find({ published: "published", enabled: true }) //
Filtrer les annonces publiées et activées .sort({ createdAt: -1 }) // Tri par
date de création, du plus récent au plus ancien .skip(offset) // Applique
l'offset pour la pagination .limit(16); // Limite le nombre de résultats à 16
annonces res.status(200).json(ads); // Renvoie les annonces dans la
réponse } catch (error) { res.status(500).json({ message: error.message });
// Renvoie une erreur en cas d'échec } };
```

- **POST /upload** : Permet l'upload de fichiers (images) pour les annonces immobilières. Les fichiers sont stockés dans un dossier local et un lien est renvoyé pour l'accès aux fichiers

### Contrôleurs (Controllers) :

Les contrôleurs sont responsables de la logique métier pour chaque route. Chaque contrôleur traite les requêtes HTTP reçues et fournit une réponse appropriée en fonction des données demandées ou envoyées.

#### Exemple de Contrôleur des annonces (Ads Controller) :

Voici un exemple du contrôleur utilisé pour récupérer les annonces immobilières. Ce contrôleur permet d'appliquer une pagination sur les annonces publiées et activées, tout en triant les résultats par date de création (les plus récentes en premier).

### Explication du fonctionnement :

```
const protectRoute = async (req, res, next) => { try {  
  const token = req.cookies.jwt ? req.cookies.jwt :  
    req.headers.token; if (!token) { return  
    res.status(401).json({ error: "Unauthorized" }); }  
  const decoded = jwt.verify(token,  
    process.env.JWT_KEY); if (!decoded) { return  
    res.status(401).json({ error: "Unauthorized" }); }  
  const user = await  
    User.findById(decoded.userId).select("-  
    password"); if (!user) { return res.status(401).json({  
    error: "Unauthorized" }); } req.user = user; next(); }  
  catch (error) { console.error("Error in protectRoute:",  
    error); res.status(500).json({ error: error.message });  
  } };
```

- **Pagination** : Le paramètre offset est utilisé pour définir à partir de quel point commencer la récupération des annonces. Cela permet de diviser les résultats en pages.
- **Filtrage** : La méthode .find() filtre uniquement les annonces qui sont publiées (published: "published") et activées (enabled: true).
- **Tri** : Les résultats sont triés en ordre décroissant en fonction de leur date de création (createdAt), pour afficher les annonces les plus récentes en premier.
- **Limite** : La méthode .limit(16) limite le nombre de résultats retournés à 16 annonces par page.

Ce contrôleur est essentiel pour garantir une navigation fluide et optimisée des utilisateurs lors de la consultation des annonces immobilières sur le site.

## Sécurité et Authentication:

La sécurité du backend repose sur l'utilisation de **tokens JWT** pour protéger les routes sensibles. Les utilisateurs doivent être authentifiés pour accéder à certaines fonctionnalités, telles que la gestion des annonces ou des profils. Un middleware est utilisé pour vérifier le token envoyé par l'utilisateur, soit via un cookie, soit via un en-tête HTTP. Voici un exemple du middleware protectRoute .

# Frontend:

## Architecture Frontend:

Le frontend de l'application a été développé en utilisant **React**, une bibliothèque JavaScript populaire pour la création d'interfaces utilisateur dynamiques et modulaires. L'application est structurée autour de composants réutilisables, ce qui facilite la maintenance et l'évolutivité. La navigation au sein de l'application est gérée par **React Router** pour permettre le chargement dynamique des pages sans rechargement complet du site. Le code est organisé de manière à séparer les composants (Navbar, ToAgent, AdminStats, etc.) et les pages (Home, SignUp, Dashboard, etc.) pour une meilleure gestion du code. Chaque page correspond à une route spécifique et utilise le composant PageTitle pour définir dynamiquement le titre de la page.

## Pages Principales du Site Web :

Voici les pages principales de votre site web avec des suggestions d'emplacements pour les captures d'écran :

### Home (Page d'accueil):

Affiche une sélection des annonces immobilières vedettes ainsi qu'une barre de recherche pour filtrer les biens. L'utilisateur peut voir des annonces populaires dès l'arrivée.

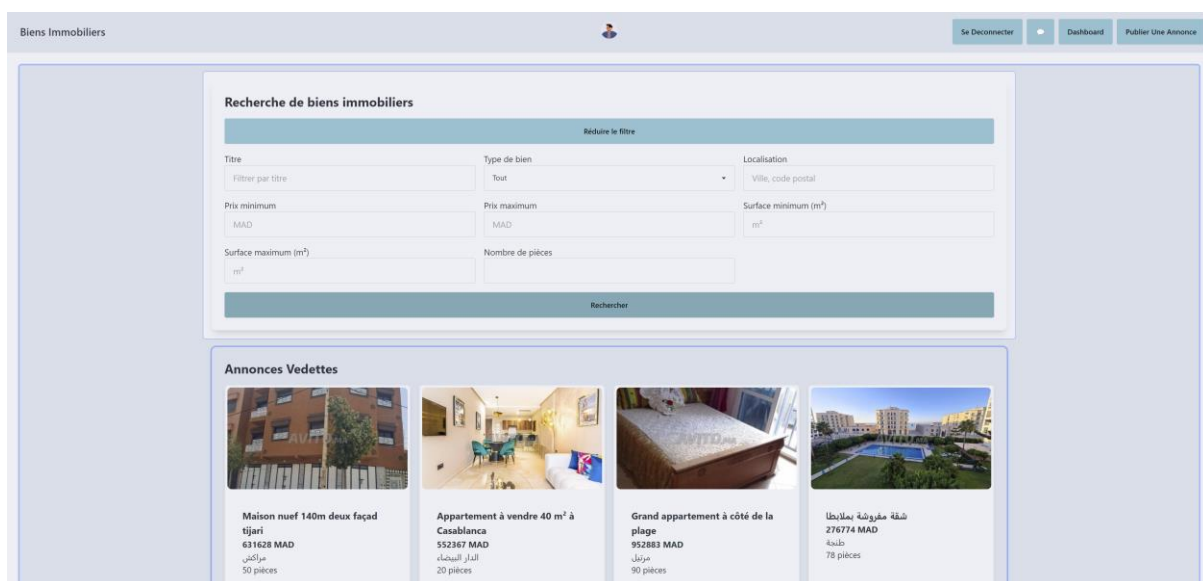


Figure 11

## SignUp (Inscription):

Page permettant aux utilisateurs de créer un compte via un formulaire contenant des champs comme prénom, nom, e-mail, mot de passe, et le rôle (agent ou utilisateur).

The screenshot shows the 'Créer un compte' (Create Account) page. At the top, there's a header with 'Biens Immobiliers' on the left and 'Se Connecter' and 'Publier Une Annonce' on the right. The main content area is titled 'Créer un compte :'. Below this, there's a purple box with an information icon and text: 'En utilisant l'authentification facile, vous pouvez remplir vos données supplémentaires ultérieurement dans les paramètres de votre compte.' The form consists of several input fields: 'Nom', 'Prénom', 'Telephone', 'Email', 'Le mot de passe', 'Confirmer le mot de passe', and a dropdown menu for 'Agent ou Utilisateur ?'. A 'Se connecter' link is located below the form. At the bottom of the form area is a 'Créer un compte' button. The footer contains four columns of links: 'Bien Immobilier' (Nous facilitons les taches de ventes et d'achats), 'SERVICES' (Branding, Design), 'COMPANY' (About us, Contact), and 'LEGAL' (Terms of use, Privacy policy).

Figure 12

## Login (Connexion):

Les utilisateurs peuvent se connecter en utilisant leurs informations d'identification.

The screenshot shows the 'Connectez-vous' (Login) page. At the top, there's a header with 'Biens Immobiliers' on the left and 'S'Inscrire' and 'Publier Une Annonce' on the right. The main content area is titled 'Connectez-vous :'. Below this, there's a purple box with an information icon and text: 'Entrez votre e-mail et mot de passe.' The form consists of two input fields: 'Email' and 'Le mot de passe'. A 'S'inscrire' link is located below the form. At the bottom of the form area is a 'Se connecter' button. The footer contains four columns of links: 'Bien Immobilier' (Nous facilitons les taches de ventes et d'achats), 'SERVICES' (Branding, Design, Marketing, Advertisement), 'COMPANY' (About us, Contact, Jobs, Press kit), and 'LEGAL' (Terms of use, Privacy policy, Cookie policy).

Figure 13



## Dashboard des non administrateur (Tableau de Bord):

Le tableau de bord personnalisé où les utilisateurs (agents ou non) peuvent voir les annonces qu'ils ont postées et gérer leurs informations.

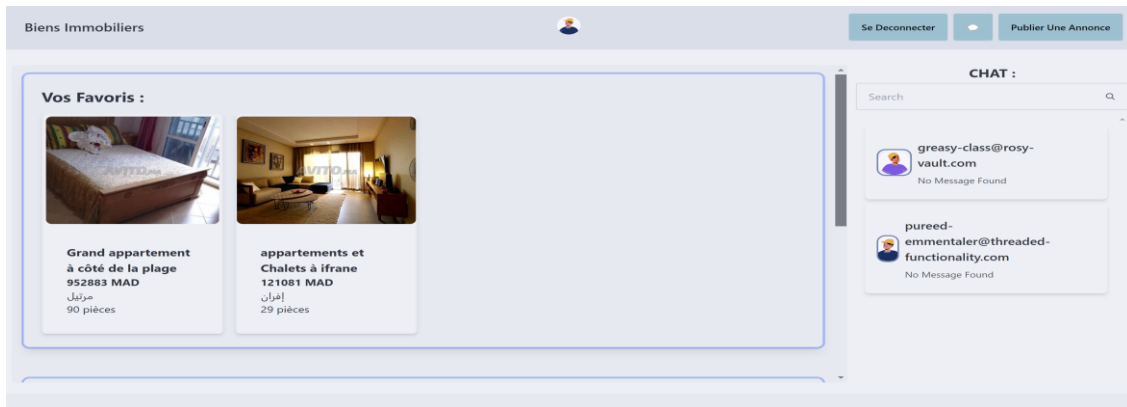


Figure 14



Figure 15

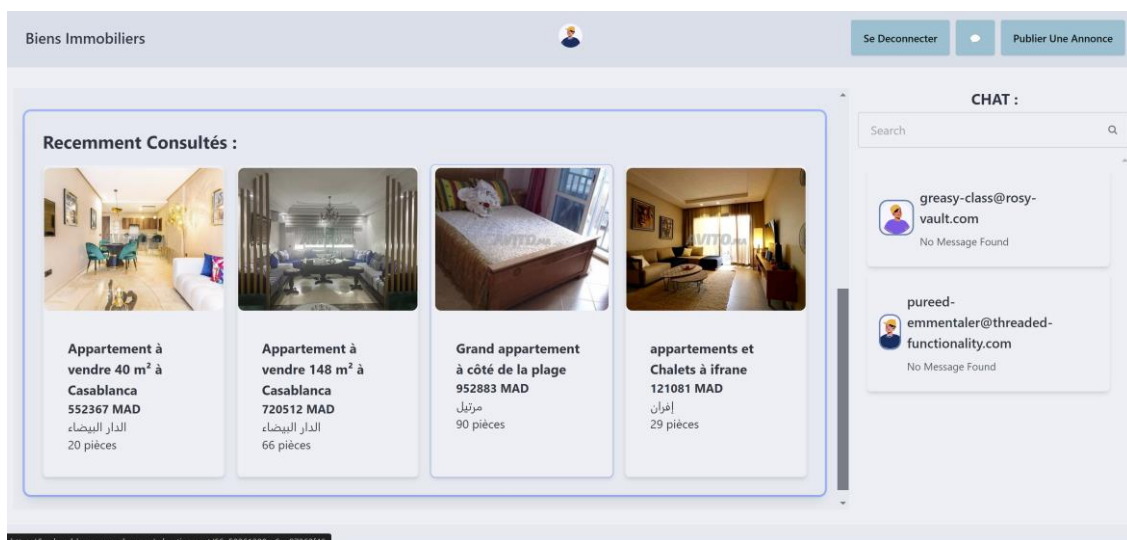


Figure 16

## InsertAd (Insertion d'une annonce):

Page permettant de créer une nouvelle annonce avec des informations comme le prix, la description, des photos, etc.

The form is titled "Qu'annoncez-vous aujourd'hui ?" and includes a sub-header "Grâce à ces informations les acheteurs peuvent trouver votre annonce plus facilement". It contains several input fields: "Type de bien", "Titre De L'Annonce", "Adresse du bien", and "Description". Below these are three smaller fields: "Nombre de pièces", "Surface", and "Prix". A "Status de publication" section follows, with "Diagnostics" and "Équipements" as options. A large image upload area is present with a red 'X' icon and the text "Insérer une image du droite". To the right of this area is a button labeled "Ajouter une photo". At the bottom right is a button labeled "Publier l'annonce".

Figure 17

## EditAd (Modifier une annonce):

Permet aux agents de modifier une annonce existante.

The form is titled "Modifier votre annonce :" and includes a sub-header "Grâce à ces informations, les acheteurs peuvent trouver votre annonce plus facilement". It contains several input fields: "Appartement", "Appartement à vendre 40 m² à Casablanca", "الدوار البيضاء", and "Appartement à vendre 40 m² à Casablanca". Below these are three smaller fields: "20", "325", and "552967". A "Publiée" section follows, with "Appartement à vendre 40 m² à Casablanca" and "Appartement à vendre 40 m² à Casablanca" as options. A large image upload area is present with a red 'X' icon and the text "Insérer une image du droite". To the right of this area is a button labeled "Ajouter une photo". At the bottom right is a button labeled "Mettre à jour l'annonce".

Figure 18

## AdDetails (Détails de l'annonce) :

Affiche les détails complets d'une annonce, incluant le prix, la description, les photos, etc.

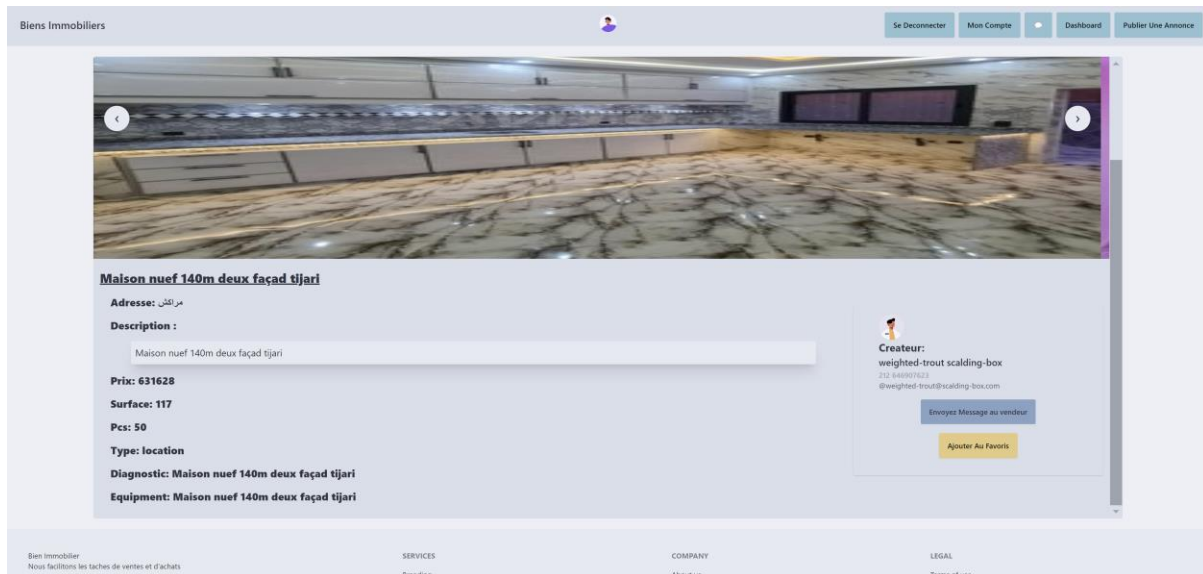


Figure 19

## Chat (Messagerie):

Système de messagerie pour permettre aux utilisateurs et agents de communiquer à propos d'annonces spécifiques.

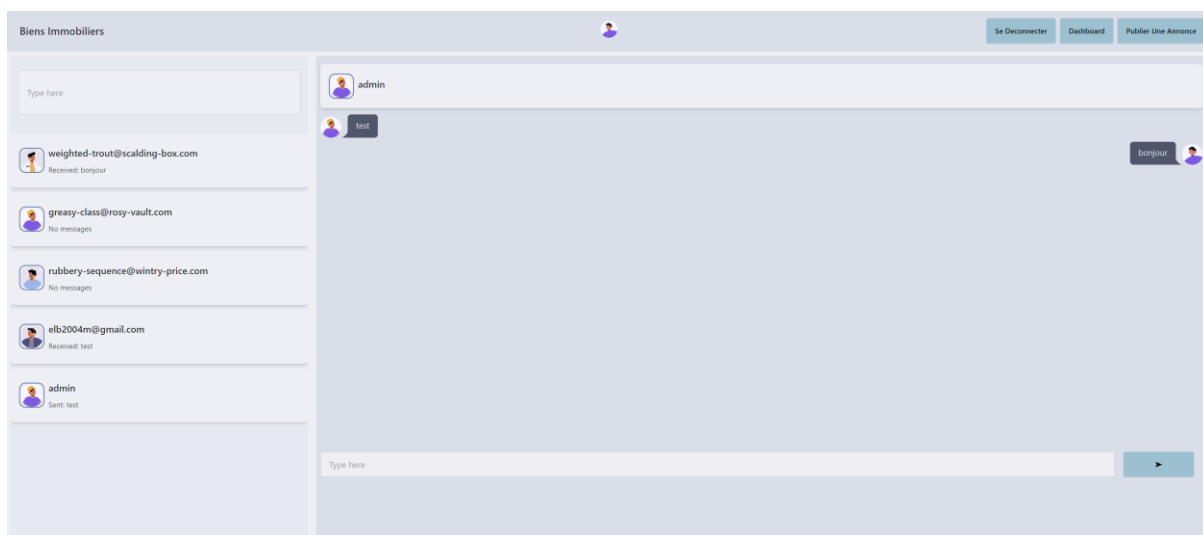


Figure 20

AdminDashboard (Tableau de bord Administrateur) :

Pour les administrateurs du site, permettant de gérer les utilisateurs, les annonces, et d'accéder à des statistiques.

AdminStats (Statistiques Administratives) :

Sous-composant du tableau de bord administrateur, affichant les statistiques globales (utilisateurs actifs, annonces, etc.).

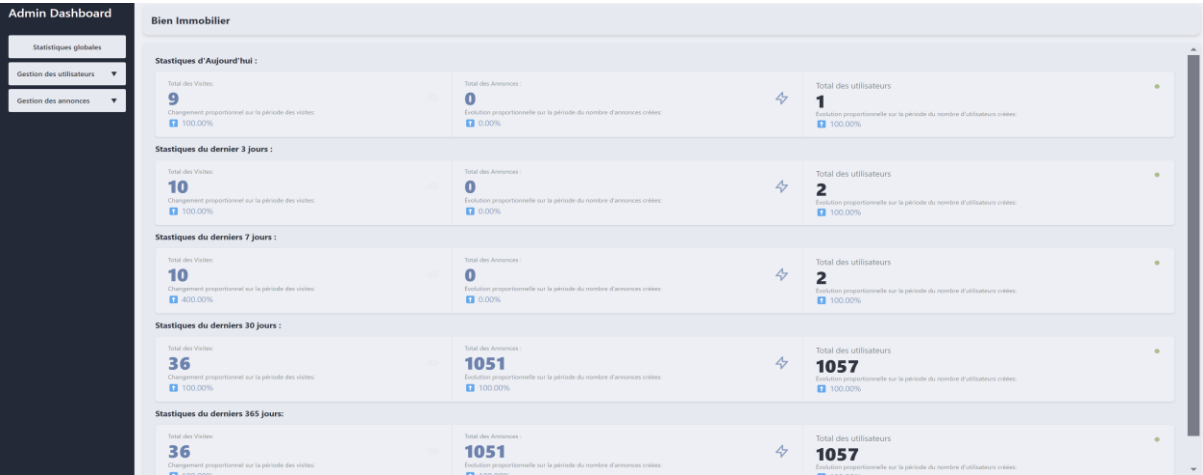


Figure 21

AdminUserList (Liste des utilisateurs):

Page permettant aux administrateurs de voir et gérer la liste des utilisateurs inscrits.

The screenshot shows the 'AdminUserList' section of the 'Admin Dashboard'. It features a sidebar with navigation options: 'Statistiques globales', 'Gestion des utilisateurs', 'Liste des utilisateurs', 'Ajouter un utilisateur', and 'Gestion des annonces'. The main content area displays a table of users with columns for profile picture, first name, last name, email, role, and a 'Supprimer' button. The table is titled 'Bien Immobilier' and contains 15 rows of user data.

Profile Picture	First Name	Last Name	email	role	Action
	courty-france	pale-chablis	courty-france@pale-chablis.com	agent	Supprimer
	quadratic-margarine	rubbery-producer	quadratic-margarine@rubbery-producer.com	agent	Supprimer
	citron-duck	distinct-hound	citron-duck@distinct-hound.com	agent	Supprimer
	tomato-sandwich	salty-pool	tomato-sandwich@salty-pool.com	agent	Supprimer
	offline-agressance	brilliant-mode	offline-agressance@brilliant-mode.com	agent	Supprimer
	serious-decibel	glad-pit	serious-decibel@glad-pit.com	agent	Supprimer
	primordial-tum	juvenile-crew	primordial-tum@juvenile-crew.com	agent	Supprimer
	ivory-arrowsnot	ordered-porica	ivory-arrowsnot@ordered-porica.com	agent	Supprimer
	flavorful-laser	irregular-tanker	flavorful-laser@irregular-tanker.com	agent	Supprimer
	grilled-thumbail	pleasant-gauls	grilled-thumbail@pleasant-gauls.com	agent	Supprimer
	kind-ridge	lively-front	kind-ridge@lively-front.com	agent	Supprimer
	honest-integrator	brilliant-chest	honest-integrator@brilliant-chest.com	agent	Supprimer
	brass-apes	minty-pomice	brass-apes@minty-pomice.com	agent	Supprimer
	spry-diffuse	bright-incubator	spry-diffuse@bright-incubator.com	agent	Supprimer

Figure 22

### AdminAddUser (Ajout d'utilisateur):

Permet à l'administrateur d'ajouter de nouveaux utilisateurs directement.

The screenshot shows the 'Admin Dashboard' on the left with a sidebar menu containing 'Statistiques globales', 'Gestion des utilisateurs', 'Liste des utilisateurs', 'Ajouter un utilisateur', and 'Gestion des annonces'. The main content area is titled 'Bien Immobilier' and features a form titled 'Ajouter un compte :'. The form includes a purple informational box stating: 'En utilisant l'authentification facile, vous pouvez remplir vos données supplémentaires ultérieurement dans les paramètres de votre compte.' Below this, the form has input fields for 'Nom', 'Prénom', 'Telephone', 'Email', 'Le mot de passe', and 'Confirmer le mot de passe'. A dropdown menu labeled 'Admin , Agent ou Utilisateur ?' is at the bottom of the form, followed by an 'Ajouter Le compte' button.

Figure 23

### AdminAdsList (Liste des annonces) :

Page permettant aux administrateurs de gérer toutes les annonces du site.

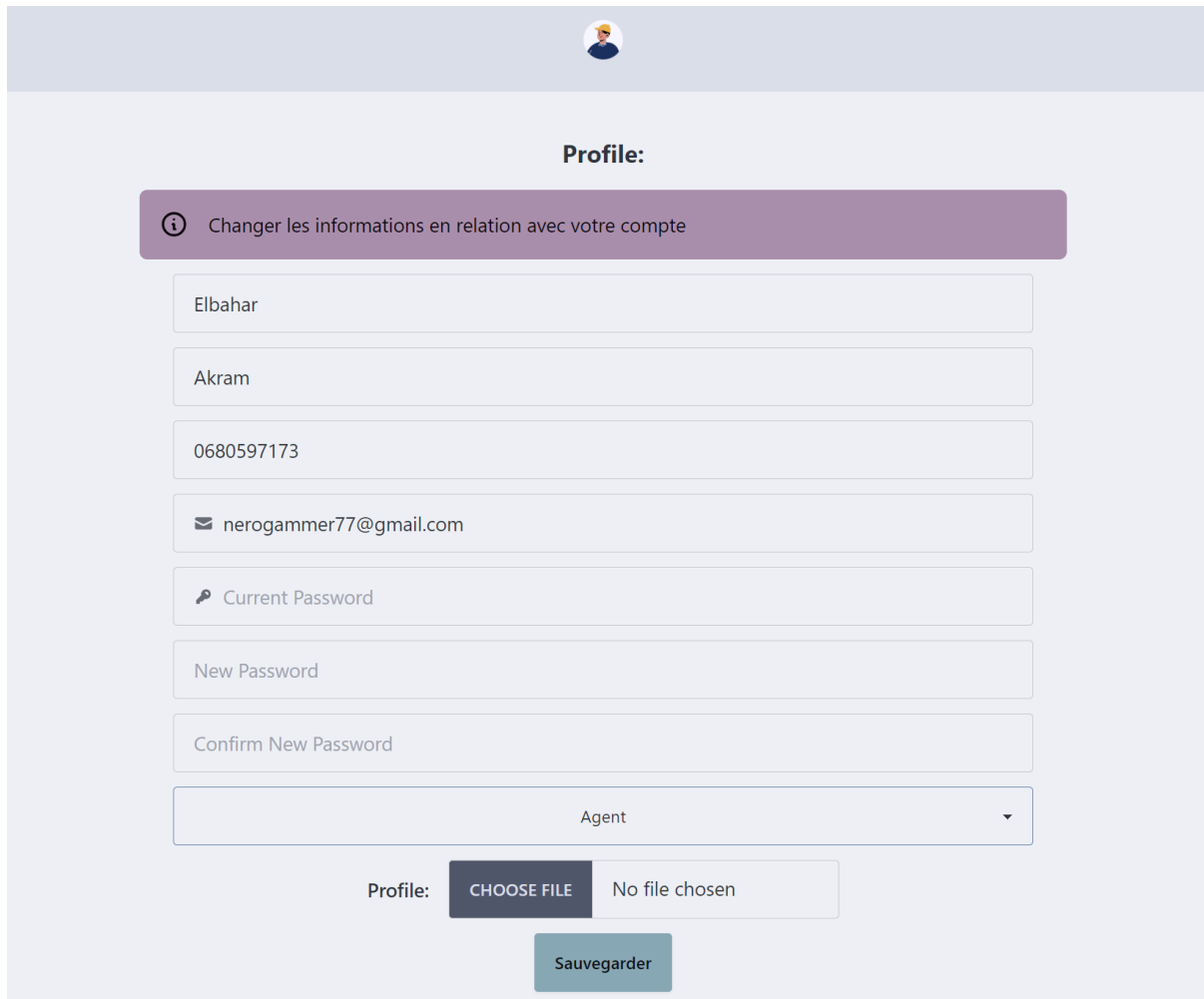
The screenshot shows the 'Admin Dashboard' on the left with a sidebar menu containing 'Statistiques globales', 'Gestion des utilisateurs', 'Liste des utilisateurs', 'Ajouter un utilisateur', and 'Gestion des annonces'. The main content area is titled 'Bien Immobilier' and displays a table of real estate listings. The table has columns for '#', 'Titre', 'Adresse', 'Prix', 'Surface', 'Type', 'Date De Publication', 'Createur', 'Vues', 'Status', and 'Actions'. The table contains 16 rows of data, each representing a different real estate listing. The 'Actions' column for each row contains a 'Supprimer' button.

#	Titre	Adresse	Prix	Surface	Type	Date De Publication	Createur	Vues	Status	Actions
2	Appartement de haut standing à Mabilif	الدار البيضاء	805572	395	achat	14/06/2024	favorful-laser@regular-tanlier.com	0	Désactiver	Supprimer
3	appartement de vacances	مراكش	272296	88	location	14/06/2024	charcoal-core@edible-wahut.com	0	Désactiver	Supprimer
4	VENTE MAGNIFIQUE VILLA A ALLIANCE MEDHA	مهدية	425667	255	location	14/06/2024	tomato-sandwich@safty-pool.com	0	Désactiver	Supprimer
5	Luxe Appartements à Témara - Courte & Longue Durée	تطوان	317493	276	vente	14/06/2024	brass-xpex@minty-pumice.com	1	Désactiver	Supprimer
6	Appart de vacances familial	مراكش	246833	212	vente	14/06/2024	primordial-tom@juvenile-cirex.com	0	Désactiver	Supprimer
7	Appartement à vendre 70 m² à Fes	فاس	168825	481	location	14/06/2024	seasoned-agent@impulsive-residual.com	0	Désactiver	Supprimer
8	studio vide	أكادير	350536	57	achat	14/06/2024	grilled-thumbnaill@pleasant-graile.com	0	Activer	Supprimer
9	Appartement Moderne à Maarif Casablanca	الدار البيضاء	717999	65	autre	14/06/2024	red-headout@cheesy-chicken.com	0	Désactiver	Supprimer
10	Appartement Au Centre Ville Avec Vue Sur Plage	طنجة	6469	142	achat	14/06/2024	chrom-duck@distinct-hound.com	0	Désactiver	Supprimer
11	Appartement 3 Chambres Quartier La Vacon, Kenitra	الكنيطرة	316688	193	autre	14/06/2024	acrylic-infection@versilean-banylax.com	0	Activer	Supprimer
12	Très bel appartement à louer	الدار البيضاء	856469	443	location	14/06/2024	serious-decibel@glad-pit.com	0	Désactiver	Supprimer
13	شقة قريبة من البحر	مراكش	196059	205	autre	14/06/2024	spry-diffuse@bright-incubator.com	0	Désactiver	Supprimer
14	VENTE APPARTEMENT 121 M2 LA VILLE HAUTE KENITRA	الكنيطرة	961583	106	location	14/06/2024	offline-agreance@brilliant-mode.com	0	Désactiver	Supprimer
15	Terrain à vendre pour villa de vos rêves	بنسليمان	632569	431	autre	14/06/2024	baked-cinnamon@parallel-grow.com	0	Désactiver	Supprimer
16	Lots de terrain villa 600m² à vendre à Calfonie	الدار البيضاء	703647	234	autre	14/06/2024	ivory-arrowroot@ordered-porrico.com	0	Activer	Supprimer

Figure 24

## EditProfile (Modification de profil):

Permet aux utilisateurs de modifier leurs informations personnelles (nom, e-mail, etc.) et d'ajouter une photo de profil.



Profile:

Changer les informations en relation avec votre compte

Elbahar

Akram

0680597173

nerogammer77@gmail.com

Current Password

New Password

Confirm New Password

Agent

Profile: CHOOSE FILE No file chosen

Sauvegarder

Figure 25

## LogOut (Déconnexion) :

Page qui effectue la déconnexion de l'utilisateur en réinitialisant les tokens d'authentification.

## Style et UI avec Tailwind CSS :

L'application utilise **Tailwind CSS** pour styliser les composants et garantir une interface utilisateur réactive et moderne. Le framework permet de rapidement construire des interfaces sans trop de CSS personnalisé, ce qui assure une grande flexibilité et rapidité dans la conception.

## Base de Données:

La base de données utilisée pour ce projet est **MongoDB**, une base de données NoSQL qui permet de stocker des documents JSON-like. Les schémas ont été définis à l'aide de **Mongoose**, une bibliothèque qui permet de structurer et de valider les données. Voici les principaux modèles de données utilisés dans le projet.

### Schéma des Annonces (Advertisement Schema) :

Ce schéma représente une annonce, généralement pour des biens immobiliers ou des produits.

- **title** (String, required) : Le titre de l'annonce.
- **description** (String, required) : La description détaillée du bien ou de l'objet mis en vente ou en location.
- **enabled** (Boolean, default: false, required) : Indique si l'annonce est activée ou désactivée.
- **published** (String, enum: ["published", "draft"], default: "draft", required) : Le statut de publication de l'annonce (publiée ou brouillon).
- **price** (Number, required) : Le prix de l'annonce.
- **surface** (Number, optional) : La surface (par exemple, en mètres carrés pour un bien immobilier).
- **pcs** (Number, optional) : Le nombre de pièces (par exemple, pour une maison ou un appartement).
- **type** (String, optional) : Le type d'annonce (immobilier, produit, etc.).
- **adresse** (String, required) : L'adresse ou la localisation de l'annonce.
- **pictures** ([String], default: [], required) : Un tableau d'URL des images associées à l'annonce.
- **diagnostic** (String, optional) : Détails des diagnostics techniques ou des évaluations.
- **equipment** (String, optional) : Équipements ou caractéristiques spécifiques du bien ou de l'objet.
- **seen** ([String], default: [], required) : Liste des utilisateurs ou IP ayant vu l'annonce.
- **createdBy** (ObjectId, ref: "User", required) : Référence à l'utilisateur ayant créé l'annonce.

## Schéma des Conversations (Conversation Schema) :

Ce schéma est utilisé pour stocker les conversations entre plusieurs participants.

- **participants** ([ObjectId], ref: "User", required) : Tableau des utilisateurs participant à la conversation.
- **messages** ([ObjectId], ref: "Message", default: []) : Tableau des identifiants des messages échangés dans la conversation.

## Schéma des Vues (Seen Schema) :

Ce schéma permet de suivre quelles annonces ont été vues par quels utilisateurs ou quelles adresses IP.

- **ad** ([ObjectId], ref: "Advertisement", required) : Référence à l'annonce vue.
- **ip** (String, required) : L'adresse IP de la personne ayant vu l'annonce.

## Schéma des Messages (Message Schema) :

Ce schéma représente un message envoyé entre utilisateurs via le système de messagerie intégré.

- **senderId** (ObjectId, ref: "User", required) : Référence à l'utilisateur qui a envoyé le message.
- **receiverId** (ObjectId, ref: "User", required) : Référence à l'utilisateur qui a reçu le message.
- **messageBody** (String, required) : Le contenu du message.



## Schéma des Utilisateurs (User Schema) :

Ce schéma représente les utilisateurs du système, qu'il s'agisse de clients, d'agents ou d'administrateurs.

- **firstName** (String, required) : Le prénom de l'utilisateur.
- **lastName** (String, required) : Le nom de famille de l'utilisateur.
- **username** (String, unique, required) : Le nom d'utilisateur unique.
- **email** (String, unique, required) : L'adresse e-mail de l'utilisateur.
- **password** (String, minlength: 5, required) : Le mot de passe de l'utilisateur (haché).
- **tel** (String, optional) : Le numéro de téléphone de l'utilisateur (facultatif).
- **role** (String, enum: ["user", "agent", "admin"], default: "user", required) : Le rôle de l'utilisateur dans le système.
- **gender** (String, enum: ["male", "female", "none"], required) : Le sexe de l'utilisateur.
- **profile\_pic** (String, default: "", optional) : L'URL de la photo de profil de l'utilisateur.
- **ads** ([ObjectId], ref: "Advertisement", default: []) : Tableau d'annonces créées par l'utilisateur.
- **seen** ([ObjectId], ref: "Advertisement", default: []) : Tableau d'annonces vues par l'utilisateur.
- **favorite** ([ObjectId], ref: "Advertisement", default: []) : Tableau des annonces favorites de l'utilisateur.

## Défis et Solutions:

Durant le développement de l'application, plusieurs défis ont été rencontrés à différents niveaux, notamment en termes de gestion de la base de données, d'intégration frontend/backend, et de performance. Voici un aperçu des principaux problèmes

### Problèmes rencontrés:

**Problème d'authentification sécurisée** : Lors de la mise en place de l'authentification avec **JWT**, la sécurisation des tokens a posé un problème, notamment pour la gestion des tokens dans les cookies pour protéger les routes sensibles.

**Pagination inefficace** : Avec une augmentation du nombre d'annonces dans la base de données, le chargement des pages est devenu plus lent, car toutes les annonces étaient récupérées en une seule requête.

### Solutions apportées:

**JWT et Cookies sécurisés** : Pour résoudre le problème d'authentification, les tokens JWT ont été stockés de manière sécurisée dans les cookies. Cela a renforcé la sécurité des tokens, et un middleware de vérification des tokens a été ajouté pour protéger les routes sensibles.

**Pagination optimisée** : Pour résoudre les lenteurs dues au chargement des annonces, une pagination a été implémentée dans les requêtes MongoDB en utilisant les méthodes `.skip()` et `.limit()`. Cela a permis de charger les annonces par blocs, améliorant ainsi la performance.

## Conclusion:

### Bilan du stage:

Ce stage a été une opportunité exceptionnelle pour appliquer les compétences acquises durant ma formation dans un projet concret et professionnel. J'ai eu l'occasion de travailler avec des technologies modernes telles que **Node.js**, **React**, **MongoDB**, et **Tailwind CSS**, ce qui m'a permis de comprendre les différentes facettes du développement web, de l'architecture backend à l'interface utilisateur.

Le projet m'a également permis de renforcer mes compétences en gestion de projet, en travaillant sur des fonctionnalités complexes tout en respectant les délais impartis.

### Acquis techniques et professionnels:

**Technologies Backend** : Maîtrise de l'environnement **Node.js** et du framework **Express.js** pour la création d'API RESTful et la gestion de la sécurité des utilisateurs avec **JWT** et **bcrypt**.

**Frontend dynamique avec React** : Capacité à développer des interfaces utilisateur interactives avec **React**, en utilisant des composants réutilisables et la gestion d'état à l'aide de **hooks**.

**Base de données NoSQL** : Expérience avec **MongoDB** pour la gestion des données, ainsi que des techniques d'optimisation telles que l'indexation et la pagination.

**Gestion des fichiers** : Capacité à gérer le téléchargement, la compression et la validation des fichiers d'images pour assurer un stockage optimisé.

# Bibliographie:

Express.js Documentation : <https://expressjs.com/>

MongoDB Documentation : <https://www.mongodb.com/>

React Documentation : <https://reactjs.org/>

Node Js Documentation : <https://nodejs.org/docs/latest/api/>

Code source du projet en github : <https://github.com/Akramelbahar/BACKEND>

Lien du site web pour test : <https://backend-hgsc.onrender.com/>