

HOOMIES IN TOWN

APPLICATION D'ASSISTANCE POUR
AIDER LES ÉTUDIANTS ÉTRANGERS
À TROUVER FACILEMENT UN
LOGEMENT ADAPTÉ À LEURS
BESOINS

PROJET RÉALISÉ DANS LE CADRE DE
LA PRÉSENTATION AU

TITRE PROFESSIONNEL DÉVELOPPEUR WEB ET WEB MOBILE

PRÉSENTÉ PAR

CHIMÈNE JUNIOR

ADRAR - DEV31 - 24 - 01 - 2024

REAC DWWM

N°	ACTIVITÉS TYPES	CP	COMPÉTENCES PROFESSIONNELLES
1	DÉVELOPPER LA PARTIE FRONT-END D'UNE APPLICATION WEB ET WEB MOBILE SÉCURISÉE	1	INSTALLER ET CONFIGURER SON ENVIRONNEMENT DE TRAVAIL EN FONCTION DU PROJET WEB OU WEB MOBILE
		2	MAQUETTER DES INTERFACES UTILISATEUR WEB ET WEB MOBILE
		3	RÉALISER DES INTERFACES UTILISATEUR STATIQUES WEB ET WEB MOBILE
		4	DÉVELOPPER LA PARTIE DYNAMIQUE DES INTERFACES UTILISATEUR
2	DÉVELOPPER LA PARTIE BACK-END D'UNE APPLICATION WEB ET WEB MOBILE SÉCURISÉE	5	METTRE EN PLACE UNE BASE DE DONNÉES RELATIONNELLE
		6	DÉVELOPPER DES COMPOSANTS D'ACCÈS AUX DONNÉES SQL ET NOSQL
		7	DÉVELOPPER LES COMPOSANTS MÉTIER CÔTÉ SERVEUR
		8	DOCUMENTER LE DÉPLOIEMENT D'UNE APPLICATION DYNAMIQUE WEB OU WEB MOBILE

DP = DOSSIER PROFESSIONEL

M = MÉMOIRE

SOMMAIRE

REAC DWWM	2
SOMMAIRE	3
LISTE DES COMPÉTENCES COUVERTES PAR LE PROJET FIL ROUGE	5
I. Développer la partie front - end d'une application web et web mobile en intégrant les recommandations de sécurité	5
A. Maquetter une application	5
B. Réaliser une interface utilisateur web statique et adaptable	5
C. Développer une interface utilisateur web dynamique	5
II. Développer la partie back - end d'une application web et web mobile en intégrant les recommandations de sécurité	6
A. Créer une base de données relationnelle	6
B. Développer la partie back - end d'une application web et web mobile	6
C. Documenter l'hébergement d'une application dynamique web ou web mobile	6
INTRODUCTION	7
RÉSUMÉ DU PROJET	8
I. What's Hoomies in town ?	8
II. Planification	9
CAHIER DES CHARGES	10
I. Contexte et Objectifs du Projet	10
A. Présentation de l'entreprise	10
B. Besoin	10
C. Objectif du site	11
D. Principaux concurrents	11
E. Cible	12

SWOT	13
CHARTE GRAPHIQUE ET LOGO	14
SPÉCIFICATION FONCTIONNELLES	15
I. Description des cas d'utilisation	15
II. Diagramme UML des cas d'utilisation	15
SPÉCIFICATIONS TECHNIQUES	17
I. Langauge et technologies	17
II. Navigateurs compatibles	18
III. Possibilité d'hébergement	18
ARBORESCENCE	20
MODÉLISATION	21
I. Wireframes	21
A. La maquette fonctionnelle	21
B. Version mobile de la page d'accueil (Mobile first)	21
II. Mockup	22
DÉVELOPPEMENT FRONT - END	23
I. Responsive	23
II. Fonctionnalité front - end	24
CONCEPTION	29
I. Modèle conceptuel des données	29
II. Modèle logique des données	29
III. PhpMyAdmin (MYSQL)	33
STRUCTURE DU PROJET MVC	35
SÉCURITÉ	35
DÉVELOPPEMENT BACK - END	37
I. Fonctionnalité back - end	37
VEILLE TECHNOLOGIQUE	39
CONCLUSION	41
REMERCIEMENT	42
ANNEXE	43

LISTE DES COMPETENCES COUVERT PAR LE PROJET

I - DÉVELOPPER LA PARTIE FRONT - END D'UNE APPLICATION WEB ET WEB MOBILE SÉCURISÉE

A - Maquetter une application

En amont de la réalisation de notre application, nous avons choisi dans un premier temps de travailler sur la réalisation des documents de conception.

Ensuite, nous avons réalisé en premier lieu des personas pour définir très clairement ce que pourront faire nos futurs utilisateurs sur notre application. Puis nous avons réalisé les wireframes du projet, aux formats mobile et desktop pour imaginer la structuration de nos pages, puis enfin nous avons réalisé une charte graphique et des maquettes pour voir concrètement à quoi allait ressembler notre application.

B - Réaliser une interface utilisateur web statique et adaptable

Ce projet est conçu comme un outil quotidien facilitant la recherche de colocation et la gestion des interactions entre colocataires potentiels. Étant donné la diversité des utilisateurs à la recherche d'une colocation, notamment en termes d'âge et de compétences numériques, il est essentiel d'offrir une interface simple et intuitive, accessible à tous, quel que soit leur niveau de maîtrise en informatique.

De plus, cet outil étant conçu pour être utilisé à tout moment et en toute mobilité, il est indispensable qu'il soit compatible avec les formats d'écran mobile, permettant ainsi aux utilisateurs de rechercher, échanger et organiser des visites en toute simplicité, où qu'ils soient.

C - Développer une interface utilisateur web dynamique

En se basant sur les profils utilisateurs définis au début du projet, on a compris qu'il fallait une interface fluide et réactive pour notre application de recherche de colocation. C'est pourquoi on a choisi d'utiliser la librairie Leaflet, qui nous a permis d'ajouter une carte interactive.

Grâce à elle, les utilisateurs peuvent facilement voir où se trouvent les différentes offres de colocation et filtrer leur recherche selon les lieux qui les intéressent.

II. DÉVELOPPER LA PARTIE BACK - END D'UNE APPLICATION WEB ET WEB MOBILE SÉCURISÉE

A - Mettre en place une base de données relationelle

Tout comme la réalisation du maquettage de l'application, nous avons voulu dès le début organiser la façon dont allaient être structurées nos données. Nous avons commencé par réfléchir aux données que nous souhaitions intégrer à l'application, puis nous avons créé un MCD, un MLD, ainsi qu'un dictionnaire des données. Pour gérer la base de données de notre application, nous avons choisi d'utiliser MYSQL.

B - Développer la partie back-end d'une application web ou web mobile

Pour réaliser le back-end de notre application en PHP, nous avons structuré notre logique serveur avec différentes routes, chacune reliée à des contrôleurs. Nous avons également mis en place des systèmes pour gérer l'authentification des utilisateurs et la validation des données.

C - Documenter l'hébergement d'une application dynamique web ou web mobile

Nous présentons différents types d'hébergement pour une application web et nous expliquons pourquoi nous avons choisi l'hébergement mutualisé.

INTRODUCTION

About me

At the age of 20, I had to end my professional football career after a serious car accident that prevented me from continuing to play at a high level. Despite this setback, my passion for sports remained intact. After much research and reflection, I decided to turn towards the field of data and now aspire to become a Data Engineer. My goal is to apply my future technical skills to the world of high-level sports, contributing in a different way to the universe that still inspires me so deeply.

CHIMÈNE JUNIOR

Work

- 2024 -- DWWM Training - Adrar
- 2021-2023 -- Space Integrator - Comat
- 2018-2021 -- Retail Sales - Footlocker
- 2011-2017 -- Semi-professional Footballer



PROJECT SUMMARY

WHAT'S HOOMIES IN TOWN ?

AFFORDABLE PREMIUM HOUSING FOR STUDENTS

The concept of Hoomies in Town is to offer high-end shared accommodations at affordable prices.

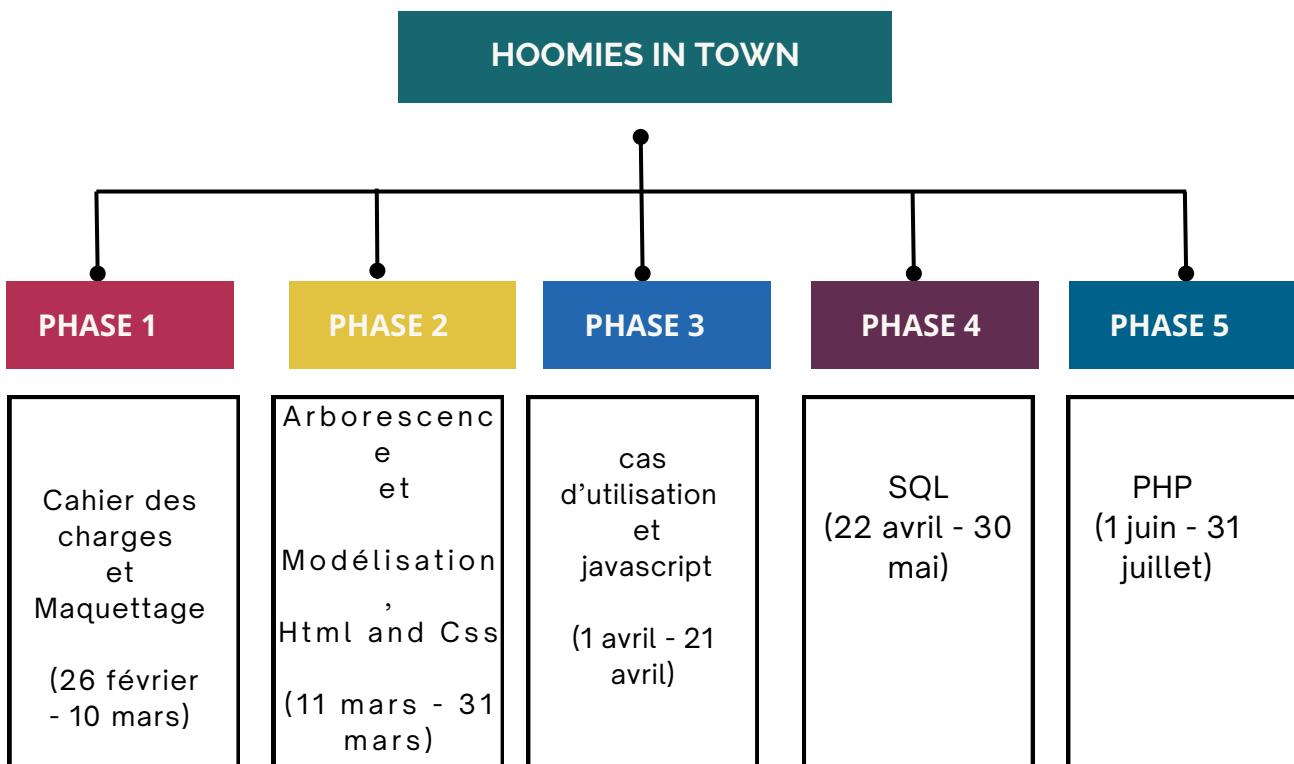
The goal is to address common problems faced by these students, such as language barriers, lack of a guarantor, and unfamiliarity with the French real estate market.

CONNECTING OWNERS AND STUDENTS

The project aims to create a platform that will connect property owners with foreign students looking for shared housing.

The accommodations offered will be carefully selected to ensure a high level of comfort while remaining within a price range accessible to students.

PLANIFICATION



La planification mise en place pour le projet Hoomies in Town s'est déroulée en cinq phases.

Les trois premières phases ont été consacrées au développement du front-end :

- La première phase a été dédiée à la rédaction des cahiers des charges et à la création des maquettes.
- La deuxième phase s'est concentrée sur l'arborescence, la modélisation, ainsi que le développement en HTML et CSS.
- La troisième phase a porté sur la définition des cas d'utilisation et l'ajout de dynamisme au projet.

Les deux dernières phases ont été consacrées au back-end :

- La phase 4 a été dédiée à SQL.
- La phase 5 s'est concentrée sur PHP.

CAHIER DES CHARGES

I. CONTEXTE ET OBJECTIFS DU PROJET

A - PRÉSENTATION DE L'ENTREPRISE

Hoomies in Town, est en cours de création , nos visons à simplifier la recherche de colocation pour les étudiants étrangers.

B - BESOIN

Les étudiants étrangers cherchent des colocataires pour plusieurs raisons principales :

1 - Réduction des coûts : Partager un logement permet de diviser le loyer et les charges, ce qui est souvent plus économique qu'un appartement individuel.

2 - Soutien social : Vivre avec d'autres personnes aide à combattre l'isolement et favorise les échanges , ce qui est important pour s'adapter à un nouveau pays.

3 - Simplicité : Les colocataires sont souvent déjà meublées et prêtées à l'emploi, ce qui facilite l'installation pour un étudiant arrivant de l'étranger.

4 - Proximité des universités : Souvent, les colocataires sont situées près des campus, ce qui réduit le temps de trajet.

C - OBJECTIF DU SITE

L'OBJECTIF PRINCIPAL EST :

Faciliter la mise en relation entre étudiants étrangers et propriétaires.

Assurer la transparence des prix et la sécurité des transactions.

D - PRINCIPAUX CONCURRENTS

LES PRINCIPAUX CONCURRENTS SUR LE MARCHÉ FRANÇAIS DE LA COLOCATION SONT :

- Appartager :

C'est une plateforme bien établie qui propose un grand choix d'annonces de colocation avec des options de filtres personnalisés, très prisée par les étudiants et jeunes actifs.

- La Carte des Colocs :

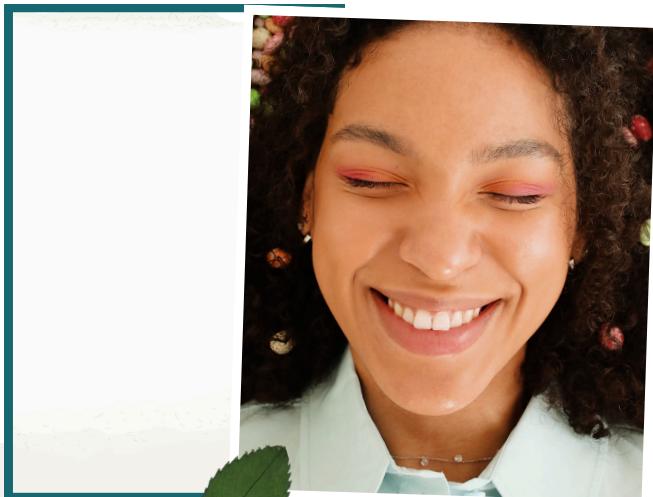
Un site plus communautaire et gratuit, avec une approche conviviale et simple, qui met l'accent sur la transparence entre colocataires grâce à des annonces locales.

- Roomlala :

Spécialisé dans la location de chambres chez l'habitant ou en colocation, ce site est souvent utilisé par des expatriés ou des voyageurs pour des séjours de courte à moyenne durée.

E - CIBLE

Afin d'avoir une bonne représentation de la cible de nos utilisateurs, nous avons établi un persona :



Hello,
I'm joy

14 102 11999

Age, 25 ans

situation: Jeune cadre
dynamique

Localisation souhaitée:
Bordeaux ↗

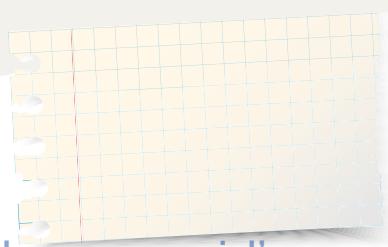


Bio

Joy Boyomo, une jeune cadre dynamique originaire d'Afrique du Sud, vient d'être mutée à Bordeaux. Malgré ses moyens financiers, elle opte pour la colocation. Ce choix est motivé par deux raisons principales : faciliter son intégration dans sa nouvelle ville et partager les frais de logement.

Attentes

- Prix : 650 EUROS
- Colocataires : Mixte
- Localisation : Bordeaux



Pour plus de persona, voir l'annexe.(P 45)

SWOT

STRENGTHS

S

WEAKNESSES

W

- Notre projet offre de la colocation haut de gamme à des prix abordables,
- Notre projet doit se construire une réputation avec des ressources financières limitées,

OPPORTUNITIES

O

T

THREATS

- 40 % des étudiants étrangers n'arrivent pas à trouver de logements, ce qui crée une demande de plus en plus forte pour des solutions comme la colocation.

- Notre projet va devoir affronter une forte concurrence, une demande qui varie, et des règles qui changent souvent, ce qui rend la gestion plus difficile.

<https://WWW.SENAT.FR/question/base/2023/qSEQ23050677.html>
ce lien mène vers l'article.

Pour plus d'information sur l'article , voir annexe (p 46)

LA CHARTE GRAPHIQUE

A - LOGO

Notre objectif était de créer une identité visuelle simple, authentique , reflétant notre marque tout en laissant une impression durable sur nos utilisateurs.



Hoomies in Town combine l'idée de "homies" (amis proches) avec le concept de colocation, conviviale .

B - COULEURS



Notre palette de couleurs a été spécialement choisie pour être accessible aux personnes atteintes de daltonisme, privilégiant des contrastes marqués et des combinaisons facilement distinguables.

C - POLICES

NANUM MYEONGJO – POLICE PRINCIPALE

A B C D E F G H I J K L M N
O P Q R S T U V W X Y Z
1 2 3 4 5 6 7 8 9 10

RALEWAY - sous - police

A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z
1 2 3 4 5 6 7 8 9 10

Nous avons choisi les polices de notre site en tenant compte des besoins des personnes dyslexiques. Ces polices, avec des caractères clairs et bien espacés, améliorent la lisibilité et rendent notre contenu accessible à tous les visiteurs.

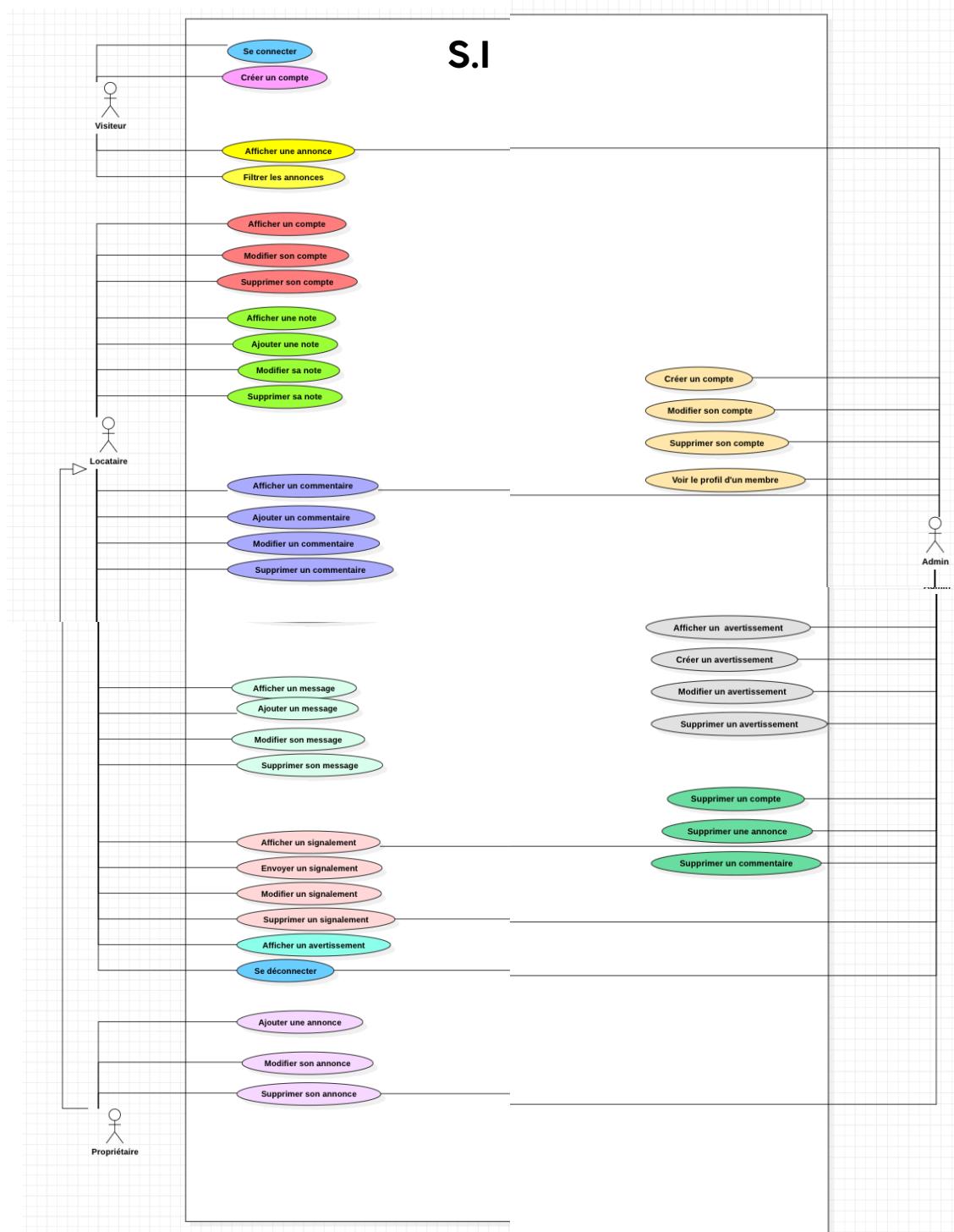
SPÉCIFICATION FONCTIONNELLES

CAS D'UTILISATION

A - DESCRIPTION DES CAS D'UTILISATION

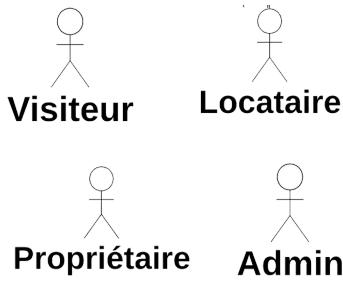
Les cas d'utilisations représentent les différentes façons dont un utilisateur peut interagir avec un système. Il schématise toutes les fonctionnalités de notre site.

B - DIAGRAMME UML DES CAS D'UTILISATION



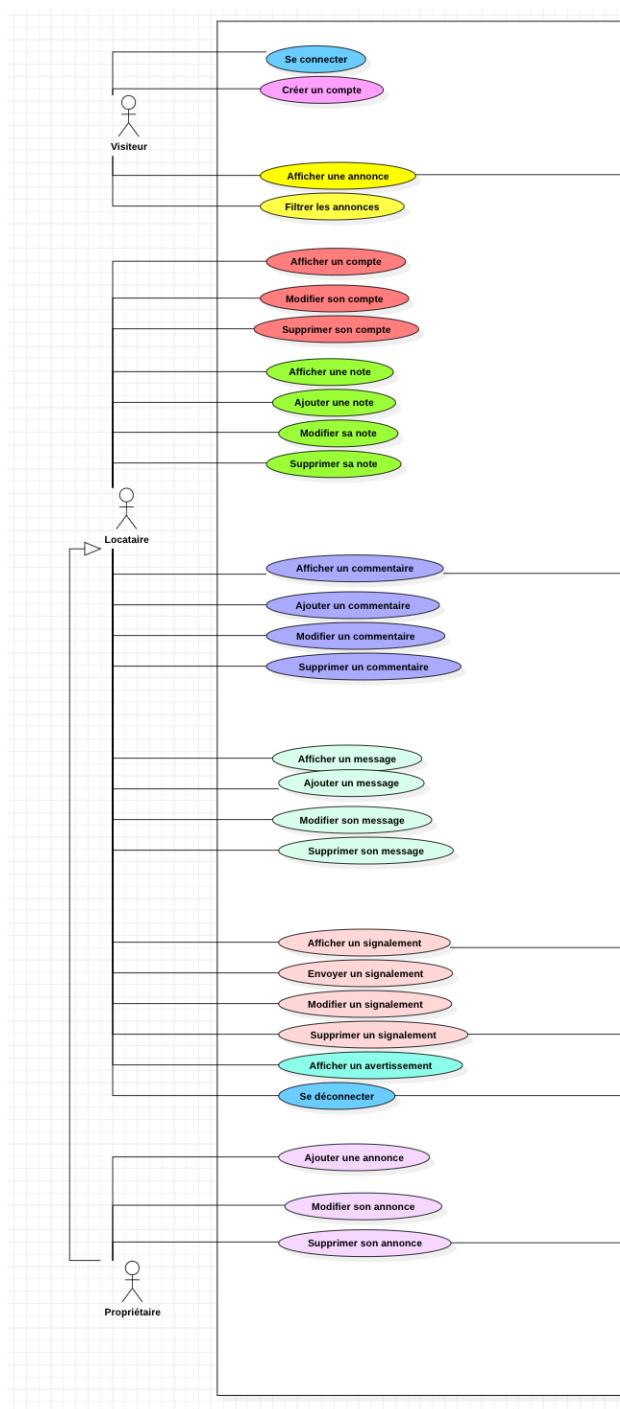
A - ACTEURS

Un acteur symbolise toute entité, humaine ou non (individu, organisation, machine, système externe), qui interagit avec l'application en cours de développement.



- Ce projet implique 4 acteurs :
- 1 - Visiteur
 - 2 - Locataire
 - 3 - Propriétaire
 - 4 - Admin

B - GÉNÉRALISATION



Cette structure montre une relation spéciale entre des acteurs, qu'on appelle relation d'héritage. Elle aide à rendre les diagrammes plus simples en évitant trop de détails. Cette relation est utile quand un acteur partage certaines actions avec un autre, mais aussi des actions spécifiques à lui.

L'héritage permet donc de montrer facilement ce que les acteurs ont en commun et ce qui les distingue dans un système.

Exemple :

Le propriétaire hérite des fonctionnalités du locataire. Contrairement au locataire, le propriétaire dispose également de la fonctionnalité d'ajouter une annonce.

SPÉCIFICATIONS TECHNIQUES

I. TECHNOLOGIES

Pour réaliser les spécifications techniques, nous avons tout d'abord réfléchi aux technologies que nous souhaitions utiliser pour réaliser ce projet. Puis nous nous sommes penchés sur quels navigateurs nous souhaitons que l'application soit utilisable de manière optimale.

Ensuite , nous avons également réfléchi à la manière dont nous pourrions déployer cette application.

A. OUTILS TECHNIQUES UTILISÉS



- HTML5, CSS3 pour la structure et le style.



- JavaScript pour l'interactivité.



- Leaflet (carte interactive)



- PHP pour la logique serveur.



- MySQL pour la base de données

B. SERVEUR



- MAMP permet de configurer un serveur local avec Apache, offrant ainsi un environnement pour développer et tester des sites web dynamiques en local avant de les déployer en ligne.

C. OUTILS DE DÉVELOPPEMENT



- Visual Studio Code comme éditeur de code.



- GitHub est une plateforme qui permet aux développeurs de stocker, gérer et partager du code source,

D. AUTRES



- Outils de design comme canva et Figma pour les maquette et wireframes.

II. NAVIGATEURS COMPATIBLES

Selon une étude de marché de 2022, les navigateurs les plus utilisés sont :

Sur desktop



- Chrome : 57,13 %



- Firefox : 16,36 %



- Edge : 11,74 %

Sur mobile



- Chrome : 56,03 %



- Safari : 30,15 %

Source :

[https://www.leptidigital.fr/webmarketing/parts-de-marche-navigateurs-web-10814/#:](https://www.leptidigital.fr/webmarketing/parts-de-marche-navigateurs-web-10814/#:~:text=Le%20march%C3%A9%20des%20navigateurs%20web%20en%20France%20en%202022)

III. HÉBERGEMENT

A - TYPE D'HÉBERGEMENT

Il y'a 4 types d'hébergements :

L'hébergement mutualisé , l'hébergement dédié , l'hébergement VPS , l'hébergement Cloud.

Hébergement Mutualisé : Abordable, pour les petits sites web ou les débutants.

Hébergement Dédié : Performances maximales et contrôle total, pour les grandes entreprises ou les sites à fort trafic.

Hébergement VPS : Bon compromis entre coût, performance et contrôle, idéal pour les sites de taille moyenne.

Hébergement Cloud : Flexible et évolutif, pour les sites à forte demande ou en croissance rapide.

B - QUEL EST L'HÉBERGEMENT CHOISI POUR HOOMIES IN TOWN ?

Pour l'instant, le site Hoomies in Town n'est pas encore en ligne. Mais à l'avenir, on pense choisir un hébergement mutualisé. Cela nous permettra de partager les ressources avec d'autres sites, ce qui sera plus économique et facilitera la gestion technique.

AVANTAGES

- PRIX ATTRACTIF
- ADAPTABILITÉ
- FLEXIBILITÉ

INCONVÉNIENTS

- SÉCURITÉ
- RISQUE DE SATURATION
- LIMITES DE PERSONNALISATION

C - CHOIX DU NOM DE DOMAINE

OFFRE GROUPÉE ET ÉCONOMIES

Achetez une offre groupée et protégez votre marque.

ÉCONOMISEZ 33 %

[hoomiesintown.fr + .com + .eu](#)

30,97 €

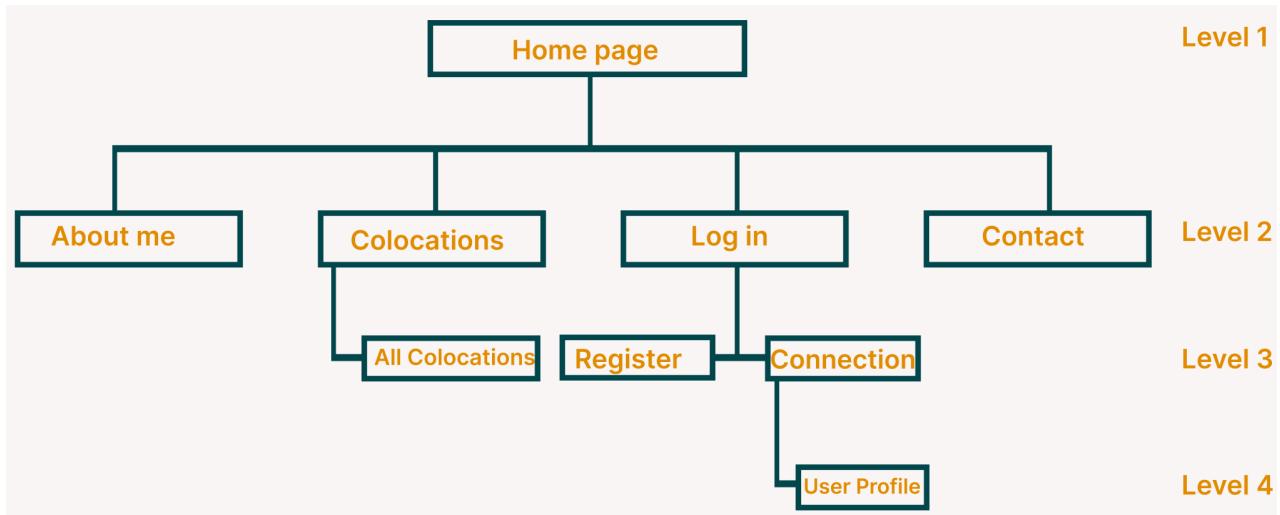
20,47 €/1ère année

[Ajouter au panier](#)

Économisez et bénéficiez d'une plus grande visibilité de votre marque grâce à une offre groupée de noms de domaine.

En combinant ces trois noms de domaine, on peut toucher à la fois un public local (fr), international (com) et européen (eu). En plus, ça protège la marque contre d'éventuelles copies ou utilisations non autorisées.

ARBORESCENCE



Cette arborescence est essentielle pour l'efficacité de notre site web. Elle permettra aux utilisateurs de naviguer facilement et de trouver rapidement l'information recherchée, ce qui va améliorer l'expérience utilisateur (UX).

Cette structure est également importante pour le référencement naturel (SEO), car les moteurs de recherche l'utilisent pour évaluer la pertinence et l'importance du contenu. Une bonne architecture augmente les chances d'apparaître en première page de Google.

Il y a également la célèbre règle des trois clics en expérience utilisateur (UX) : le principe est que les utilisateurs doivent pouvoir accéder à l'information qu'ils recherchent en trois clics maximum, faute de quoi ils risquent de quitter le site.

Pour plus d'information sur l'arborescence, voir l'annexe. (P 47)

Modélisation

I. WIREFRAME

A - LA MAQUETTE FONCTIONNELLE

Le wireframe est un schéma qui structure une page web, visualisant les zones de texte, les médias, les liens et les éléments graphiques. Essentiel pour la conception d'interfaces utilisateur, il définit la répartition des composants et sert de base pour créer l'interface finale.

Cette démarche vise à optimiser l'ergonomie et améliorer l'expérience utilisateur.

B - VERSION MOBILE DE LA PAGE D'ACCUEIL (MOBILE FIRST)

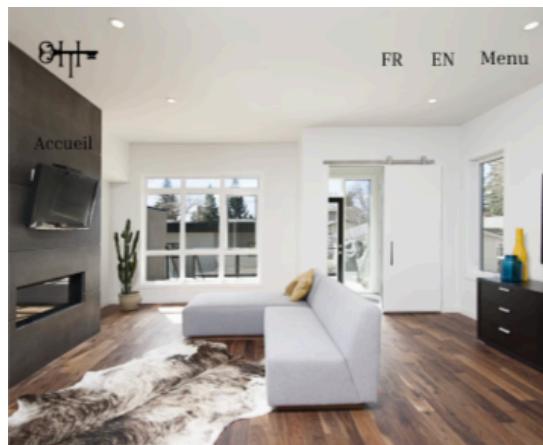


II. MOCKUP

A - VERSION MOBILE DE LA PAGE D'ACCUEIL

Le mock-up offre une représentation réaliste du futur site web ou de l'application mobile. C'est une simulation statique de l'interface utilisateur, de fidélité moyenne à élevée, réalisée avant le début du développement.

Pour la maquette, nous avons opté pour un design moderne et convivial. Nous avons choisi des couleurs tendances qui reflètent l'esprit contemporain de la colocation, tout en sélectionnant des images qui mettent en avant l'unité et la vie en communauté.



Des logements haut-de-gamme
Soucieux de garantir la pérennité de votre investissement et votre confort, tous nos logements sont entièrement meublés et équipés avec des espaces privés pour plus d'intimité.

Une présence partout en France
Chez Colocatère, la proximité est une priorité. Présents dans plus de 30 villes en France, nos équipes et nos agences vous accueillent pour échanger sur votre projet.

Hoomies In Town, les spécialistes de la colocation
Ne cherchez plus !
Votre projet d'investissement ou votre futur logement est chez nous et partout en France.

A photograph of a modern living room with teal walls, a white sofa, and a round coffee table.

You have a question ?

Besoin d'aide ? Contactez-nous ! Notre équipe est à votre disposition pour répondre à toutes vos questions et vous accompagner dans votre recherche de colocation. Nos équipes sont là pour vous aider à trouver la colocation idéale !

Envoyer un message

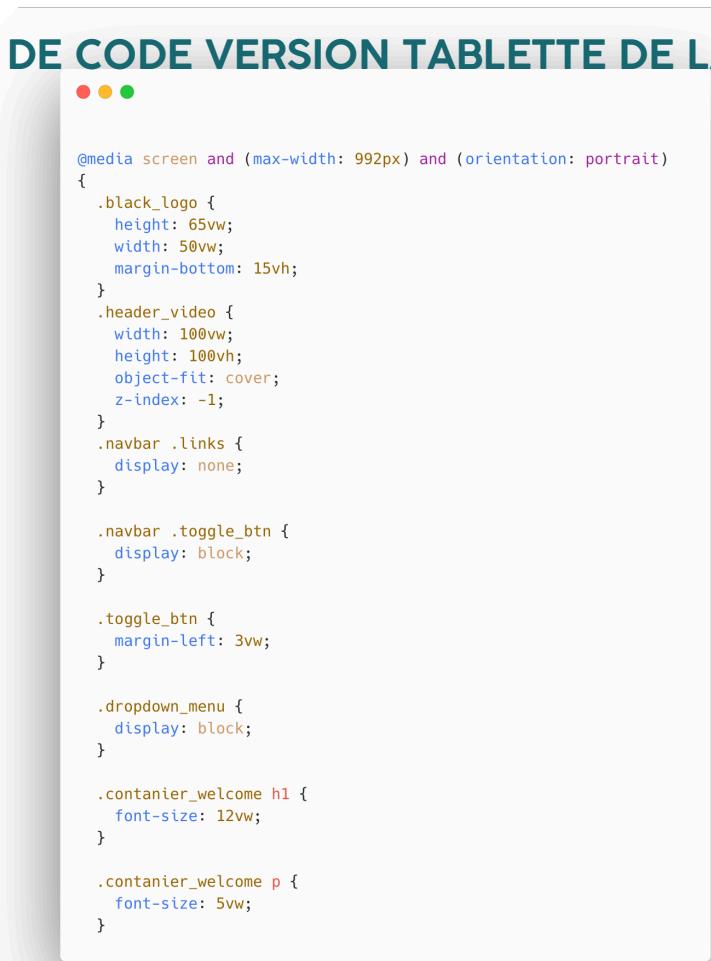
Mentions légales CGU Données personnelles et cookies
© 2024 Copyright: Hoomies in Town

DÉVELOPPEMENT FRONT - END

I. RESPONSIVE

Le site de Hoomies in Town est entièrement responsive, ce qui signifie qu'il s'adapte automatiquement à tous les types d'appareils, que ce soit sur ordinateur, tablette ou smartphone. L'expérience utilisateur reste fluide et optimale, quel que soit l'écran utilisé.

A - EXTRAIT DE CODE VERSION TABLETTE DE LA PAGE D'ACCUEIL



```
@media screen and (max-width: 992px) and (orientation: portrait)
{
    .black_logo {
        height: 65vw;
        width: 50vw;
        margin-bottom: 15vh;
    }
    .header_video {
        width: 100vw;
        height: 100vh;
        object-fit: cover;
        z-index: -1;
    }
    .navbar .links {
        display: none;
    }

    .navbar .toggle_btn {
        display: block;
    }

    .toggle_btn {
        margin-left: 3vw;
    }

    .dropdown_menu {
        display: block;
    }

    .contanier_welcome h1 {
        font-size: 12vw;
    }

    .contanier_welcome p {
        font-size: 5vw;
    }
}
```

J'ai utilisé un point de rupture (break point) à 992px, ce qui correspond généralement à la largeur d'une tablette en mode portrait. Dès que l'écran passe en dessous de cette taille, j'ai ajusté le design pour garantir une bonne expérience utilisateur.

Pour les dimensions du logo (black_logo), je les ai définies en pourcentage par rapport à la largeur de l'écran (VW), ce qui permet au logo de s'adapter naturellement à différentes tailles d'écran. De cette façon, il reste bien proportionné, même sur les petits appareils. Quant à la vidéo dans le header, je l'ai configurée pour occuper toute la largeur et la hauteur de l'écran (100vw et 100vh), tout en conservant de bonnes proportions grâce à object-fit : cover. Ainsi, elle reste toujours bien adaptée, même sur les petits écrans.

II. FONCTIONNALITÉ DU FRONT - END

A - VERSION DESKTOP DE LA PAGE COLOCATIONS

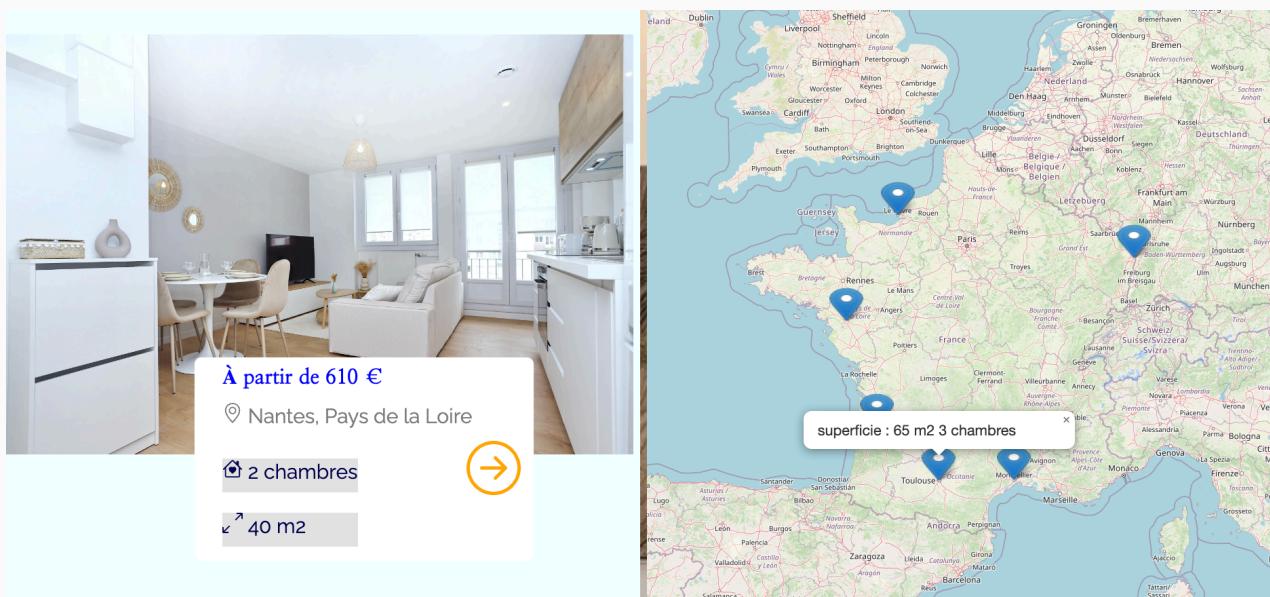
The screenshot shows the desktop version of the Hoomies in Town website. At the top, there's a search bar with placeholder text "Rechercher une chambre en colocation." Below it is a message: "Vous cherchez une chambre en colocation ? Ne cherchez plus ! Chez Hoomies in Town, nous proposons des logements en colocation offrant des espaces tout confort et entièrement équipés." A navigation bar with tabs like "Accueil", "Logements", "Services", "Blog", and "Contact" is visible. The main content area features a search form with fields for "Saisissez une ville", "Type", "Nombre de colocataires", and a "Définir la recherche" button. Below this is a grid of six apartment listings, each with a small image, location, price, number of rooms, and size. To the right of the grid is a map of France with several locations marked. The bottom of the page has a dark footer with the Hoomies logo, social media links for LinkedIn, Instagram, and Facebook, and copyright information: "En savoir plus" and "© Copyright 2024 : Chimène junior".

Pour que l'expérience utilisateur soit agréable sur notre interface, nous avons mis en avant les appartements en affichant leur prix, leur localisation, le nombre de chambres et la superficie.

Quand un utilisateur clique sur un appartement, la carte zoomé directement sur son emplacement et une popup apparaît pour le représenter.

En cliquant sur cette popup, l'utilisateur peut voir le nombre de chambres et la superficie de l'appartement.

B - OBJECTIF DE LA FONCTIONNALITÉ



Cette fonctionnalité est là pour rendre la recherche d'appartements facile et plaisante. Avec des interactions dynamiques, on aide les utilisateurs à découvrir des biens immobiliers tout en s'assurant qu'ils passent un bon moment sur le site.

B - FRONT-END CODE HTML

```
● ● ●  
  
<article id="appart6">  
    
  
  <article class="container-list4">  
    <h4>À partir de 470 €</h4>  
    <p class="localisation">  
      <svg xmlns="http://www.w3.org/2000/svg" width="30" height="30" fill="currentColor" class="bi-lts" viewBox="0 0 16 16">  
        <path d="M12.166 8.94c-.524 1.062-1.234 2.12-1.96 3.07A32 32 0 0 1 8 14.58a32 32 0 0 1 2.206-2.57c-.726-.95-1.436-2.008-1.96-3.07C3.304 7.867 3 6.862 3 6a5 5 0 0 1 10 0c .862-.305 1.867-.834 2.94M8 16s6-5.686 6-10A6 6 0 0 2 6c0 4.314 6 10 6 10"/>  
      </svg>  
      Montpellier, Occitanie  
    </p>  
  
    <p class=" cercle">  
      <svg xmlns="http://www.w3.org/2000/svg" width="80" height="80" fill="currentColor" class="bi-circle" viewBox="0 0 16 16">  
        <path fill-rule="evenodd" d="M1 8a7 7 0 0 1 0 14 0A7 7 0 0 1 8m15 0A8 8 0 1 1 0 8a8 8 0 1 1 0M4.5 7.5a.5.5 0 0 0 1h5.793l-2.147 2.146a.5.5 0 0 0 .708.708l-3a.5.5 0 0 0 -.708l-3a.5.5 0 1 0 -.708.708l10.293 7.5z"/>  
      </svg>  
    </p>  
  
    <p class=" chambre">  
      <svg xmlns="http://www.w3.org/2000/svg" width="30" height="30" fill="currentColor" class="bi-heart" viewBox="0 0 16 16">  
        <path d="M8 6.982C9.664 5.309 13.825 8.236 8 12 2.175 8.236 6.336 5.309 8 6.982"/>  
        <path d="M8.707 1.5a1 1 0 0 1-1.414 0L.646 8.146a.5.5 0 0 0 .708.707L2 8.207V13.5A1.5 1.5 0 0 0 3.5 15h9a1.5 1.5 0 0 0 1.5-1.5V8.207L.646.646a.5.5 0 0 0 .708-.707L13 5.793V2.5a.5.5 0 0 0 0-5-.5h-1a.5.5 0 0 0-.5v1.293zM13 7.207V13.5a.5.5 0 0 1-1.5h-9a.5.5 0 0 1-1.5-5V7.207L5-5z"/>  
      </svg>  
      3 chambres  
    </p>  
  
    <p class=" superficie">  
      <svg xmlns="http://www.w3.org/2000/svg" width="30" height="30" fill="currentColor" class="bi-expand" viewBox="0 0 16 16">  
        <path fill-rule="evenodd" d="M5.828 10.172a.5.5 0 0 0-.707 0l-4.096 4.096V11.5a.5.5 0 0 0 0-1v3.975a.5.5 0 0 0 0-5.5H4.5a.5.5 0 0 0 0-1H1.732l4.096-4.096a.5.5 0 0 0 0-7.07m4.344-4.344a.5.5 0 0 0 4.096a.5.5 0 0 0 0 .707"/>  
      </svg>  
      52,90 m2  
    </p>  
  </article>  
</article>  
  
<article id="map"></article>
```

Dans ce code, j'ai structuré le contenu de manière claire et sémantiquement correcte, en suivant les bonnes pratiques du balisage HTML. J'ai utilisé des icônes SVG pour améliorer l'interface utilisateur tout en gardant le design léger et flexible. J'ai également laissé de la place pour intégrer des fonctionnalités dynamiques à l'avenir, comme via JavaScript.

Je me suis assuré que le code est bien optimisé pour le SEO et l'accessibilité, en créant une hiérarchie d'informations logique. Cela facilite non seulement la compréhension pour les utilisateurs, mais aussi pour les moteurs de recherche. En plus, l'organisation des classes CSS que j'ai mise en place rend le code facile à maintenir et à faire évoluer à mesure que le projet grandit.

C - FRONT-END CODE CSS



```
#map {
    height: 100%;
    flex: 1;
}

.container-colocation {
    display: grid;
    grid-template-columns: 2fr 1fr;
}

.appartements-colocation {
    display: grid;
    grid-template-columns: 1fr 1fr;
    background-color: azure;
}

.appartements-colocation article img {
    width: 100%;
}

.appartements-colocation article img:hover {
    opacity: 0.5;
}

.container-list {
    position: absolute;
    top: 2000px;
    left: 250px;
}
```

1. La carte (#map) :

J'ai défini une hauteur de 100% pour que la carte occupe tout l'espace vertical disponible dans son conteneur. Avec l'utilisation de flex: 1, la carte s'ajuste automatiquement à l'espace restant si elle est à l'intérieur d'un conteneur flexible. Cela va garantir que la carte s'adapte parfaitement à la taille de la fenêtre ou du conteneur parent.

2. Disposition de la page avec deux colonnes (.container-colocation) :

J'ai opté pour une disposition en grille avec deux colonnes. La première colonne prend deux tiers de l'espace (avec 2fr), tandis que la deuxième prend un tiers (avec 1fr). Cela permet d'organiser efficacement la page avec, par exemple, une large section pour la carte et une plus petite pour des informations supplémentaires ou la liste des appartements.

3. Présentation des appartements en grille (.appartements-colocation) :

Les appartements sont affichés sous forme de grille à deux colonnes égales, ce qui permet de bien répartir les informations visuellement. Le fond de cette section est défini en bleu clair (azure), ce qui aide à différencier cette partie du reste de la page.

4. Images des appartements :

Les images des appartements sont ajustées pour occuper 100% de la largeur de leur conteneur, garantissant ainsi qu'elles s'affichent en pleine largeur. J'ai également ajouté un effet visuel sur les images : lorsqu'un utilisateur passe la souris dessus, l'image devient semi-transparente (opacity: 0.5), ce qui ajoute un effet interactif simple mais efficace.

5. Conteneur flottant pour la liste d'appartements (.container-list) :

Ce conteneur est positionné de manière absolue à une grande distance du haut de la page (top: 2000px), ce qui signifie qu'il est initialement hors du champ de vision. Ce choix suggère qu'il pourrait être affiché plus tard via une animation ou lorsqu'on fait défiler la page. Il est placé à 250px à partir du côté gauche, ce qui positionne précisément ce conteneur flottant.

D - FRONT-END CODE JAVASCRIPT

```
● ● ●  
const map = L.map('map').setView([48.8552, 2.3467], 6);  
const tiles = L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {  
    maxZoom: 19,  
    attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'  
}).addTo(map);
```

J'ai utilisé la bibliothèque Leaflet pour créer une carte interactive. J'ai commencé par initialiser la carte avec la méthode L.map(), qui affiche la carte dans un élément HTML. J'ai ensuite défini la vue initiale avec la méthode .setView() pour centrer la carte sur Paris et régler le zoom à un niveau modéré.

Pour afficher les tuiles (les images de la carte), j'ai utilisé la méthode L.tileLayer(), qui permet de charger les tuiles à partir de l'API OpenStreetMap. Le niveau de zoom maximal est fixé à 19, et j'ai ajouté une attribution pour respecter les droits d'auteur d'OpenStreetMap.

```
● ● ●  
let coords = [[43.600000,1.433333],[48.584614,7.7507127],  
[49.298897,0110391302],[44.841225,-0.5800364],[43.6112422,3.8767337]];  
let areas = ["65 m2", "50 m2", "42,12 m2", "40 m2", "49,19 m2", "52,90 m2"];  
let rooms = [3, 2, 3, 2, 3, 3];
```

Ensuite, j'ai un tableau coords dans lequel je stocke les coordonnées géographiques de plusieurs appartements. En parallèle, j'ai créé deux autres tableaux : areas pour stocker les superficies des appartements et rooms pour le nombre de chambres. Ces informations me permettent d'associer des détails spécifiques à chaque appartement.

```
● ● ●  
for (let i = 0; i < position; i++) {  
    let pop = L.popup({  
        CloseOnClick: true  
    }).setContent('superficie : ' + areas[i] + ' ' + rooms[i] + '  
chambres');  
    marker = L.marker(coords[i]).addTo(map).bindPopup(pop);  
}
```

Dans la boucle for, je crée un marqueur pour chaque appartement à l'aide de L.marker(). Chaque marqueur a un popup, que j'ai défini avec L.popup(), et ce popup contient les informations sur la superficie et le nombre de chambres. J'ai ensuite ajouté ces marqueurs à la carte avec la méthode .addTo(map).



```
nbAppartement[i].addEventListener("click", () =>
{   map.flyTo(coords[i], 16);
});
```

Pour l'interaction utilisateur, j'ai utilisé addEventListener() pour écouter les clics sur des éléments HTML spécifiques, chacun représentant un appartement. Quand l'utilisateur clique sur l'un d'eux, la méthode map.flyTo() est utilisée pour zoomer sur les coordonnées de cet appartement avec un niveau de zoom de 16.

E - ÉVOLUTION DU CODE

Actuellement, j'ai tout codé en dur, c'est-à-dire que les coordonnées et les informations des appartements sont directement intégrées dans le script sous forme de tableaux. Pour faire évoluer ce code, je pourrais connecter cette partie à une base de données où les informations des appartements seraient stockées.

Voici ce que je ferais pour cela :

1. Récupération des données dynamiques :

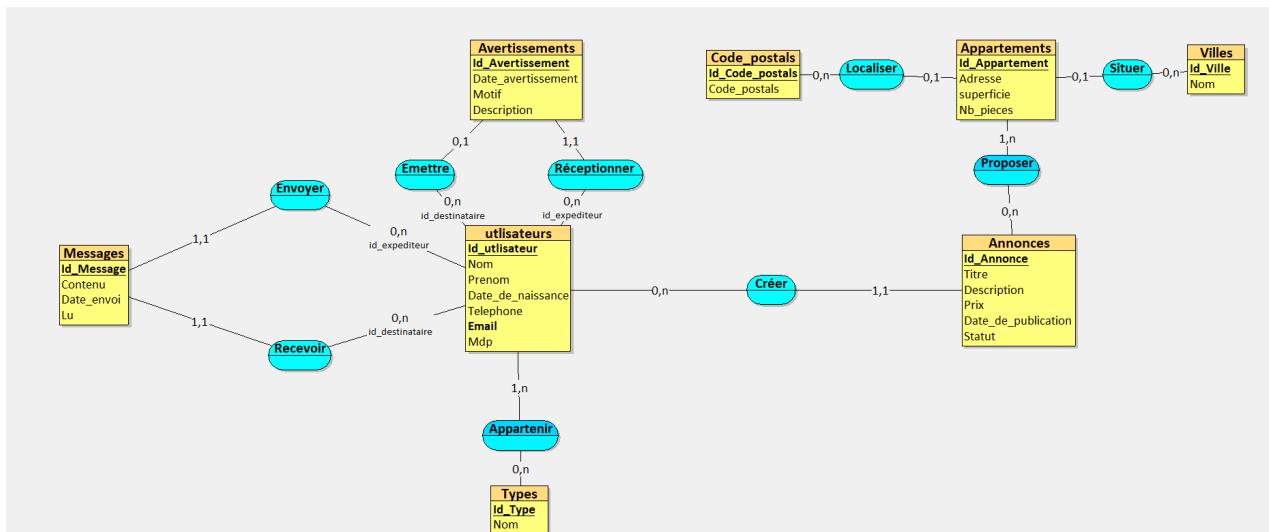
Je connecterais une base de données via une API back-end. Les informations des appartements (coordonnées, superficie, chambres) seraient récupérées via des requêtes API plutôt que d'être définies statiquement dans le JavaScript.

2. Utilisation de fetch() :

Pour charger ces données dynamiquement, je ferais des requêtes avec fetch() afin d'obtenir les coordonnées et autres détails depuis la base de données. Les données récupérées seraient ensuite utilisées pour générer les marqueurs et les popups sur la carte.

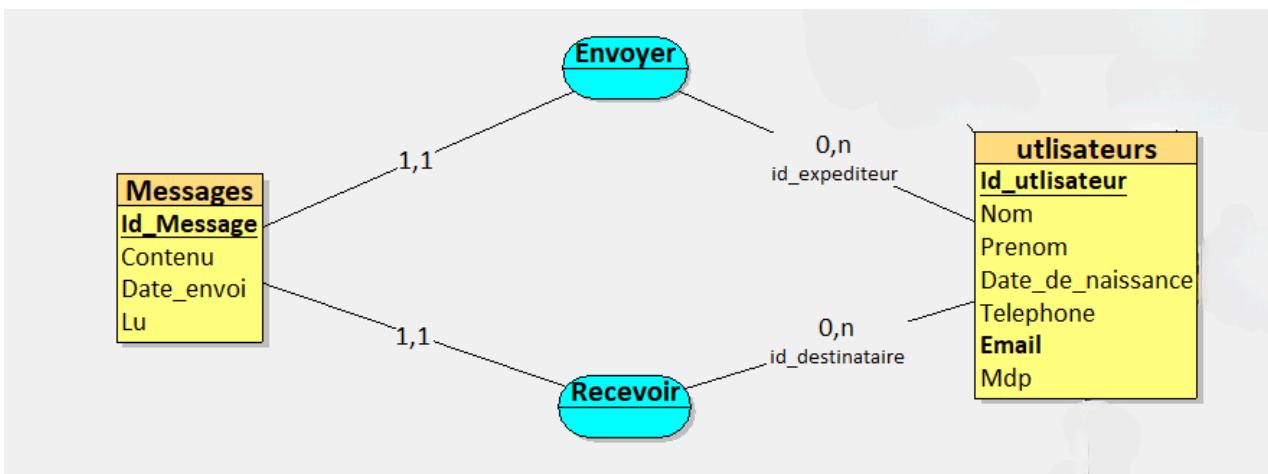
CONCEPTION

I. MODÉLE CONCEPTUELLE DES DONNÉES



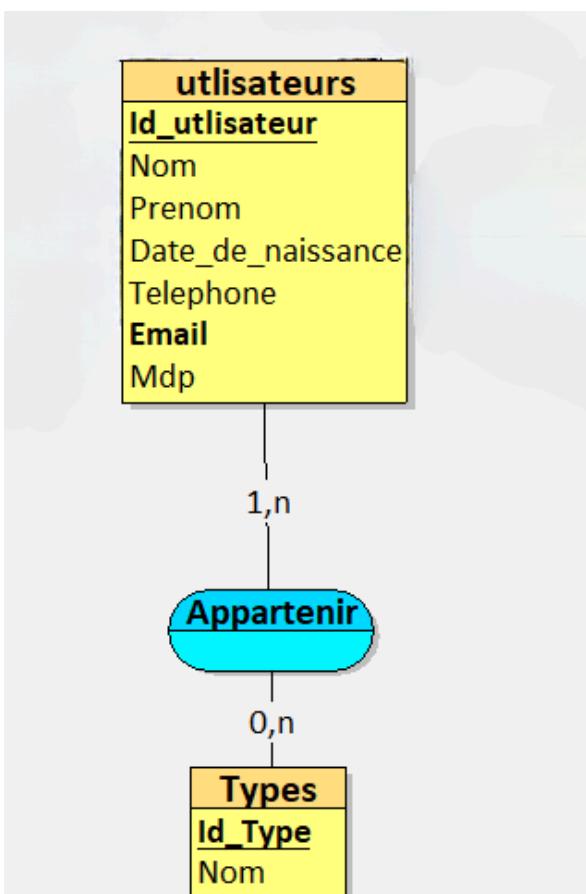
Dans le Modèle Conceptuel de Données (MCD), j'organise les données en entités, qui représentent des objets du monde réel avec leurs caractéristiques, et en associations, qui montrent les liens entre ces entités.

Les cardinalités définissent les relations possibles, comme un-à-un, un-à-plusieurs, ou plusieurs-à-plusieurs. Le MCD me permet de structurer les données et pose les bases pour créer le modèle relationnel, qui sera ensuite traduit en table dans la base de données.



EXAMPLE :

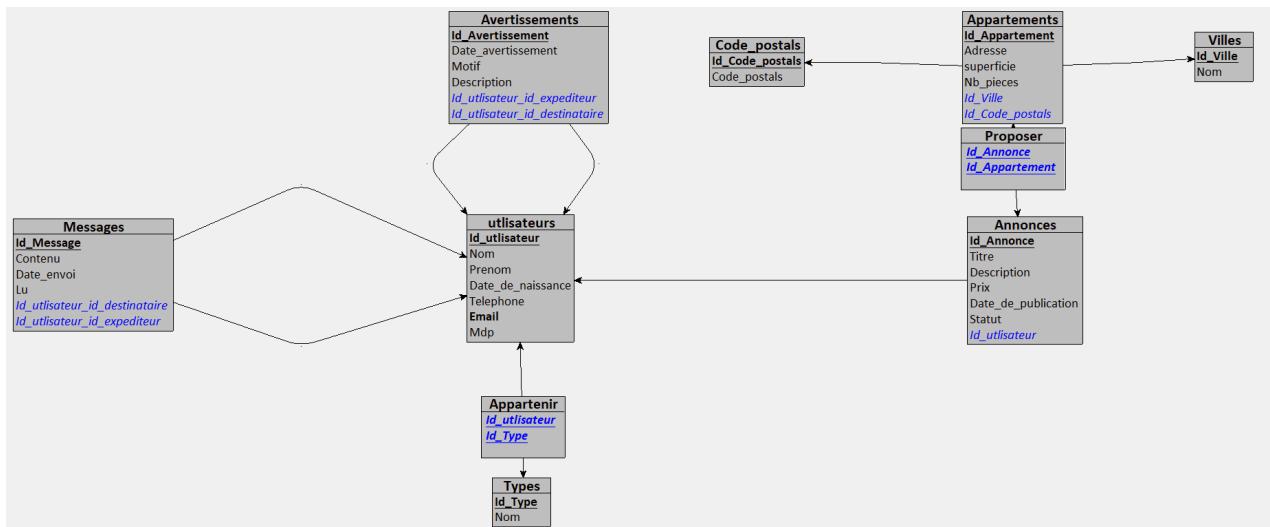
Dans mon exemple N°1, un message peut être envoyé à un seul et même utilisateurs, tandis qu'un utilisateurs peut envoyer 0 ou plusieurs messages. Ces cardinalités montrent qu'il y a un maximum de n d'un côté (utilisateur) et un maximum de 1 de l'autre côté (Messages). Cela crée une association de type 1.



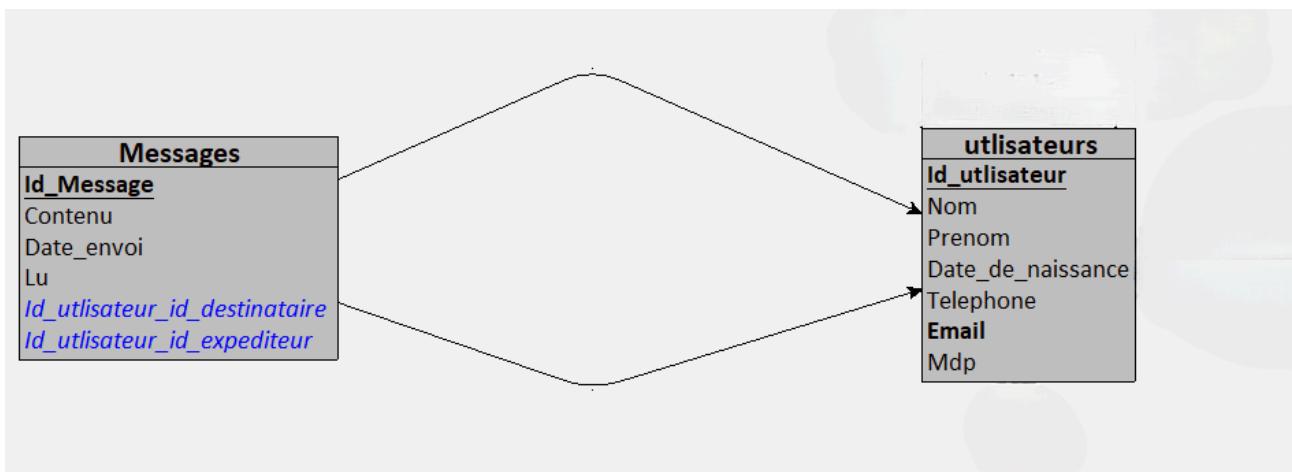
EXAMPLE :

Dans mon exemple N°2, un utilisateur peut appartenir à un seul et plusieurs types, tandis qu'un utilisateurs peut appartenir 0 ou plusieurs types. Ces cardinalités montrent qu'il y a un maximum de n d'un côté (utilisateur) et un maximum de n de l'autre côté (Types). Cela crée une association de type N.

II. MODÉLE LOGIQUE DES DONNÉES



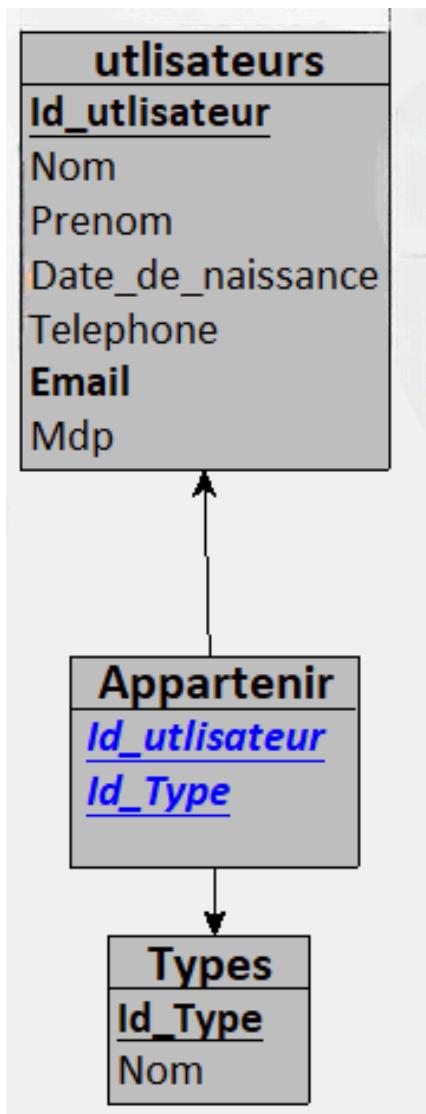
Le modèle relationnel organise les données (attributs) en "relations", qui deviennent des "tables" dans la base de données. Ce modèle repose sur le concept de dépendances fonctionnelles et est également connu sous le nom de Modèle Logique de Données (MLD).



EXEMPLE :

Dans notre Modèle Logique de Données (MLD), cette association de type 1 signifie que la clé primaire de l'entité "Utilisateur" sera ajoutée comme clé étrangère dans l'entité "Messages". Cela permet de lier chaque message à un utilisateur unique.

EXEMPLE :



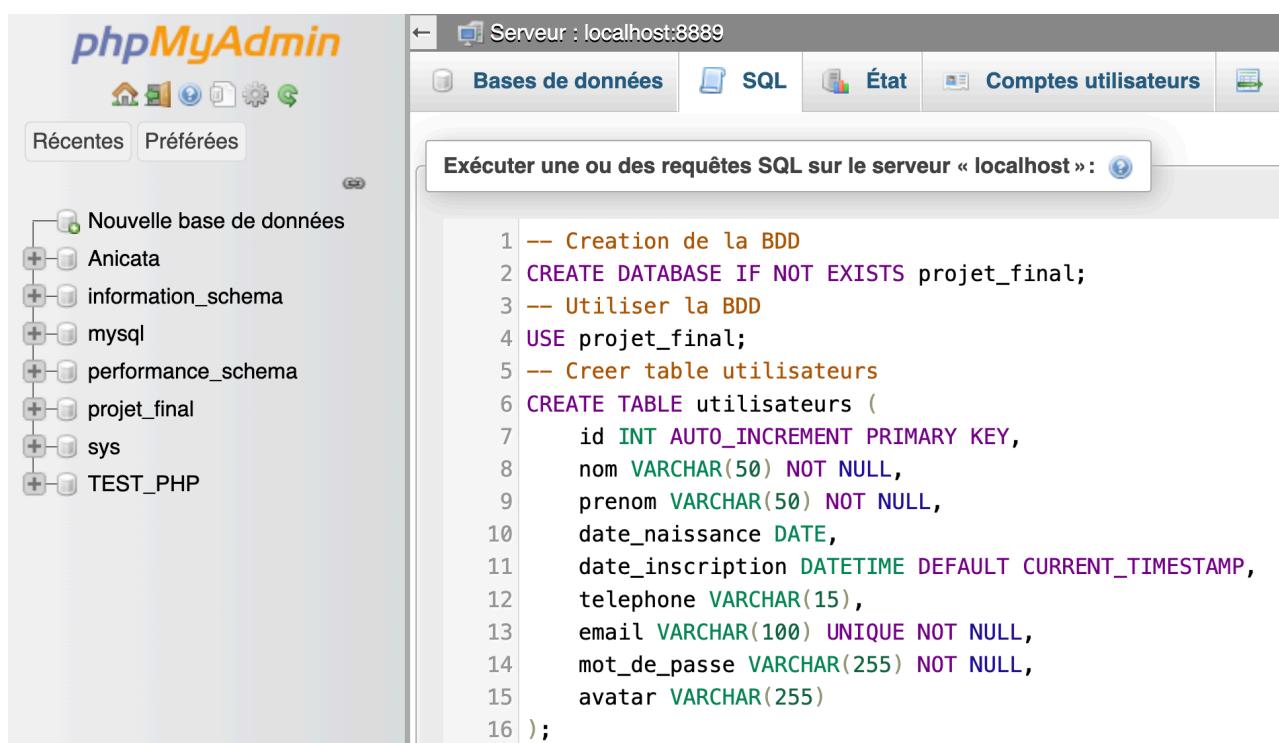
Dans notre Modèle Logique de Données (MLD), cette association de type N entraîne la création d'une table intermédiaire, souvent appelée table de jonction ou table d'association. Cette table contiendra les clés primaires des deux entités, "Utilisateurs" et "Types", qui deviennent des clés étrangères dans cette nouvelle table. Cette structure permet de gérer la relation multiple entre utilisateurs et types, en permettant à chaque utilisateur d'être lié à plusieurs Types et à chaque Types d'être liée à plusieurs utilisateurs.

PHPMYADMIN (MYSQL)

A - FONCTION DE PHPMYADMIN

PhpMyAdmin m'a permis de gérer ma base de données MySQL via une interface web. J'ai pu créer, modifier ou supprimer des bases de données, des tables et des enregistrements facilement. J'ai aussi pu exécuter des requêtes SQL, importer et exporter des données, et gérer les utilisateurs et les permissions.

B - CRÉATION DE LA BASE DE DONNÉE MYSQL



The screenshot shows the phpMyAdmin interface with the following details:

- Left sidebar:** Shows a tree view of databases: Nouvelle base de données, Anicata, information_schema, mysql, performance_schema, projet_final, sys, and TEST_PHP.
- Top menu:** Shows "Serveur : localhost:8889" and tabs for "Bases de données", "SQL", "État", and "Comptes utilisateurs".
- Central area:** A text input field titled "Exécuter une ou des requêtes SQL sur le serveur « localhost »:" containing the following SQL code:

```
1 -- Creation de la BDD
2 CREATE DATABASE IF NOT EXISTS projet_final;
3 -- Utiliser la BDD
4 USE projet_final;
5 -- Creer table utilisateurs
6 CREATE TABLE utilisateurs (
7     id INT AUTO_INCREMENT PRIMARY KEY,
8     nom VARCHAR(50) NOT NULL,
9     prenom VARCHAR(50) NOT NULL,
10    date_naissance DATE,
11    date_inscription DATETIME DEFAULT CURRENT_TIMESTAMP,
12    telephone VARCHAR(15),
13    email VARCHAR(100) UNIQUE NOT NULL,
14    mot_de_passe VARCHAR(255) NOT NULL,
15    avatar VARCHAR(255)
16 );
```

Pour créer la DATABASE , j'utilise la requête SQL suivante :

```
1 -- Creation de la BDD
2 CREATE DATABASE IF NOT EXISTS projet_final;
```

Ensuite j'ai créé la table utilisateur. J'utilise la requête SQL suivante :

```

5 -- Creer table utilisateurs
6 CREATE TABLE utilisateurs (
7     id INT AUTO_INCREMENT PRIMARY KEY,
8     nom VARCHAR(50) NOT NULL,
9     prenom VARCHAR(50) NOT NULL,
10    date_naissance DATE,
11    date_inscription DATETIME DEFAULT CURRENT_TIMESTAMP,
12    telephone VARCHAR(15),
13    email VARCHAR(100) UNIQUE NOT NULL,
14    mot_de_passe VARCHAR(255) NOT NULL,
15    avatar VARCHAR(255)
16 );

```

AUTO_INCREMENT : Permet de générer automatiquement une nouvelle valeur pour chaque nouvelle ligne, souvent utilisé pour les identifiants uniques.

VARCHAR(n) : Type de données utilisé pour stocker des chaînes de caractères de longueur variable, avec n indiquant le nombre maximal de caractères.

DEFAULT CURRENT_TIMESTAMP : Utilisé pour définir une valeur par défaut pour une colonne.

NOT NULL : Contrainte qui indique qu'une colonne ne peut pas contenir de valeurs vides.

UNIQUE : Assure que toutes les valeurs dans la colonne sont uniques dans la table.

The screenshot shows the phpMyAdmin interface with the following details:

- Serveur :** localhost:8889
- Base de données :** projet_final
- Structure :** Selected tab.
- SQL :** Tab available.
- Rechercher :** Search bar.
- Requête :** Query builder.
- Exporter :** Export options.
- Importer :** Import options.
- Opérations :** Operations menu.
- Priviléges :** Privileges menu.
- Plus :** More options.

Tables List:

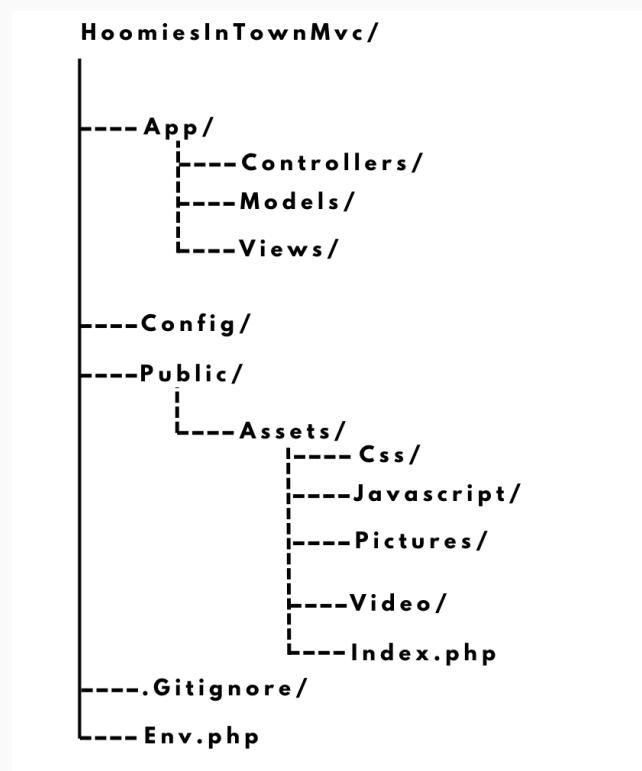
Table	Action	Lignes	Type	Interclassement	Taille
annonces	Parcourir Structure Rechercher Insérer Vider Supprimer	4	InnoDB	utf8_general_ci	32
appartements	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8_general_ci	32
appartenir	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8_general_ci	32
avertissements	Parcourir Structure Rechercher Insérer Vider Supprimer	2	InnoDB	utf8_general_ci	32
codes_postaux	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8_general_ci	32
messages	Parcourir Structure Rechercher Insérer Vider Supprimer	4	InnoDB	utf8_general_ci	48
type_users	Parcourir Structure Rechercher Insérer Vider Supprimer	3	InnoDB	utf8_general_ci	16
utilisateurs	Parcourir Structure Rechercher Insérer Vider Supprimer	6	InnoDB	utf8_general_ci	32

Database Structure:

- Nouvelle base de données
 - Anicata
 - information_schema
 - mysql
 - performance_schema
- projet_final
 - Nouvelle table
 - annonces
 - appartements
 - appartenir
 - avertissements
 - codes_postaux
 - messages
 - type_users
 - utilisateurs** (highlighted)
 - villes
 - voir

La table utilisateurs a bien été créée dans la base de données Projet_final.

STRUCTURE DU PROJET MVC



Le projet Hoomies in town est organisé selon le modèle MVC, qui facilite la maintenance et l'évolution du site.

Voici La structure des dossiers :

app/ : Contient la logique métier de l'application, y compris les contrôleurs, les modèles et les vues.

config/ : Regroupe les fichiers de configuration, tels que les paramètres de la base de données et les configurations de l'application.

public/ : Contient les fichiers accessibles publiquement, comme les fichiers CSS, JavaScript, les images , vidéo et le fichier **index.php** servant de point d'entrée.

SÉCURITÉ

En anticipant les menaces potentielles, j'ai sécurisé l'application pour protéger au mieux les données de nos utilisateurs ,

1- HACHAGE DU MOT DE PASSE

Fonction clé = password_hash()

Cette fonction est utilisée pour créer un nouveau hachage pour un mot de passe. Elle prend en compte plusieurs paramètres pour garantir un niveau de sécurité élevé ,elle utilise l'algorithme bcrypt, connu pour sa robustesse.

2- INJECTION SQL :

J'ai utilisé PDO pour sécuriser toutes mes interactions avec la base de données grâce aux requêtes préparées, ce qui me protège efficacement des attaques par injection SQL

```
public function getUserByEmail($email)
{
    $statement = $this->db->prepare("SELECT * FROM utilisateurs WHERE Email =
:email")$statement->bindParam(':email', $email);
    $statement->execute();
    return $statement->fetch(PDO::FETCH_ASSOC);
}
```

3- VÉRIFICATION DE L'INTÉGRITÉ DES DONNÉES:

Chaque fois qu'un utilisateur soumet des données, il est impératif de mettre en place des mécanismes de validation pour s'assurer qu'elles sont conformes aux spécifications et ne présentent aucun risque pour la sécurité de l'application. Pour lutter contre les injections Sql et les injections XSS.

A. Nettoyage et sécurisation des données

Pour renforcer la sécurité et empêcher l'exécution de code malveillant, les formulaires sont filtrés et vérifiés afin d'éliminer tout contenu susceptible de déclencher une attaque XSS et d'assurer la conformité des données avec les formats attendus.

- **TRIM()** : ÉLIMINE LES ESPACES INUTILES AU DÉBUT ET À LA FIN D'UNE CHAÎNE DE CARACTÈRES.
- **STRIPSLASHES()** : SUPPRIME LES ANTISLASHS AJOUTÉS POUR ÉCHAPPER DES CARACTÈRES SPÉCIAUX.
- **STRIP_TAGS()** : RETIRE TOUTES LES BALISES HTML ET PHP D'UNE CHAÎNE.
- **HTMLENTITIES()** : CONVERTIT LES CARACTÈRES SPÉCIAUX EN ENTITÉS HTML POUR UNE UTILISATION SÉCURISÉE DANS DU HTML.

FONCTIONNALITÉ DU BACK-END

Pour notre projet Hoomies in Town, j'ai voulu mettre en avant le CRUD pour la gestion des utilisateurs. Voici comment j'ai structurer chaque fonctionnalité :

1. Créer un compte :

- Les utilisateurs pourront s'inscrire en remplissant un formulaire avec leurs informations de base comme l'email, le mot de passe, et le prénom.



```
public function addUser($nom, $prenom, $date_naissance, $date_inscription, $telephone, $email, $mot_de_passe)
{
    $sql = "INSERT INTO utilisateurs (nom, prenom, date_naissance, date_inscription, telephone, email, mot_de_passe) VALUES (:nom, :prenom, :date_naissance, :date_inscription, :telephone, :email, :mot_de_passe)";
    try {
        $statement = $this->db->prepare($sql);
        $statement->execute([
            ':nom' => $nom,
            ':prenom' => $prenom,
            ':date_naissance' => $date_naissance,
            ':date_inscription' => $date_inscription,
            ':telephone' => $telephone,
            ':email' => $email,
            ':mot_de_passe' => $mot_de_passe
        ]);
    } catch (PDOException $e) {
        // Gestion des erreurs
        throw new Exception("Erreur lors de la requête : " . $e->getMessage());
    }
}
```

Cette méthode encapsule la logique nécessaire pour ajouter un nouvel utilisateur à une base de données. Elle est sécurisée car elle utilise des requêtes préparées pour éviter les injections SQL. Elle est également robuste grâce à la gestion des erreurs.

2. Afficher le compte :

- Une fois connectés, les utilisateurs pourront consulter leur profil complet avec leurs informations personnelles .

3. Modifier le compte :

- Ils pourront mettre à jour leurs informations personnelles (photo de profil, Nom, Prénom, etc.) à tout moment.

4. Supprimer le compte :

- Si un utilisateur souhaite quitter la plateforme, il pourra supprimer son compte de manière permanente. J'ai ajouter une confirmation supplémentaire pour éviter les suppressions accidentelles.

```
public function deleteUser($id)

{
    $stmt = $this->db->prepare("DELETE FROM utilisateurs WHERE id =
:id");
    $stmt->bindParam(':id', $id, PDO::PARAM_INT);

    return $stmt->execute();
}
```

- **Définition de la fonction:**

- public function deleteUser(\$id): J'ai déclaré une fonction publique nommée deleteUser qui prend en paramètre un identifiant \$id.

- **Préparation de la requête SQL:**

- \$stmt = \$this->db->prepare("DELETE FROM utilisateurs WHERE id = :id");
- \$this->db->prepare(): Cette partie prépare une requête SQL pour l'exécution.
- "DELETE FROM utilisateurs WHERE id = :id" : C'est la requête SQL elle-même. Elle demande de supprimer toutes les lignes de la table utilisateurs où la colonne id correspond à la valeur du paramètre :id. Le :id est un marqueur nommé qui sera remplacé par la valeur réelle de l'identifiant lors de l'exécution de la requête.

- **Liaison du paramètre:**

- \$stmt->bindParam(':id', \$id, PDO::PARAM_INT);: Cette ligne lie le marqueur nommé :id à la variable PHP \$id. Le troisième argument PDO::PARAM_INT spécifie que le paramètre est un entier. Cela permet d'éviter les injections SQL.

- **Exécution de la requête:**

- return \$stmt->execute();: Cette ligne exécute la requête préparée et retourne un booléen indiquant si l'exécution s'est déroulée avec succès (TRUE) ou non (FALSE).

VEILLE TECHNOLOGIQUE

Lors de ce projet, j'ai rencontré une difficulté : je savais que je voulais intégrer une carte interactive sur le site avec des fonctionnalités attractives pour nos futurs utilisateurs. Pour atteindre cet objectif, j'ai dû réaliser une veille technologique afin de trouver les meilleures solutions disponibles.

DIFFICULTÉ RENCONTRÉE

RECHERCHE DE SOLUTIONS DE CARTOGRAPHIE

- La principale difficulté de notre projet a été de visualiser précisément les emplacements des appartements sur une carte en manipulant correctement les données géographiques et les outils de cartographie.
- J'avais le choix entre Google Maps, OpenLayers, et Leaflet pour mon projet, et j'ai opté pour Leaflet en raison de sa facilité d'utilisation et de personnalisation.

R

I

RESULTAT

IMPLÉMENTATION

- Grâce à Leaflet, on a pu concevoir une interface plus intuitive et agréable. La navigation est simplifiée, et l'accès aux informations est rapide, ce qui améliore nettement l'expérience utilisateur.
- Des marqueurs indiquent l'emplacement des appartements, chaque marqueur étant associé à des informations détaillées sur l'appartement, telles que l'adresse, la description, la surface, le nombre de pièces et le type de bien.

CONCLUSION

En conclusion, je suis très satisfait du travail accompli jusqu'à présent pour ce projet de colocation. Bien que ce ne soit qu'un premier jet, je suis conscient qu'il y a encore des améliorations à apporter, notamment en fonction des nouvelles compétences et connaissances que je vais acquérir dans le futur.

Le projet devrait être finalisé d'ici la fin de l'année 2025, et je suis convaincu que les prochaines étapes permettront d'enrichir et de perfectionner le projet.

POINT D'AMÉLIORATION

Nous allons aussi renforcer la sécurité et la confiance sur notre plateforme. Pour cela, nous mettrons en place un système de vérification des profils et des annonces .

En plus, nous offrirons la possibilité de laisser des avis et des notes sur les logements et colocataires, pour rendre le tout plus transparent et rassurant pour les nouveaux arrivants.

REMERCIEMENTS

Je tiens également à exprimer ma gratitude envers l'ADRAR, qui m'a permis d'acquérir des bases solides pour la suite de mon parcours. Un grand merci à mes formateurs pour leur soutien, leurs conseils et leur expertise tout au long de cette formation.

Je remercie également mes camarades pour leur collaboration et leur esprit d'entraide, qui ont grandement contribué à cette expérience enrichissante.

ANNEXE

A - CAHIER DES CHARGES	44
B - PERSONA	45
C - ARTICLE	46
D - ARBORESCENCE	47
E - MODÉLISATION	48
• WIREFRAMES	48
• MOCKUP	50
F - MODÉLE LOGIQUE DES DONNÉES	51

LE REPOS DU PROJET EST ACCESSIBLE À CETTE ADRESSE :

<https://github.com/cammonp799/HoomiesInTownMvc/tree/master>

CAHIER DES CHARGES

CONTRAINTE TECHNIQUE

Gestion des Utilisateurs :

Sécurité des comptes et gestion des rôles. Conformité aux réglementations sur la protection des données.

Sécurité :

Contre les Attaques : Mise en place de mesures de sécurité contre les attaques courantes (SQL injection, XSS, CSRF).

Accessibilité et UX :

Design Inclusif : Interface utilisateur accessible aux personnes ayant des handicaps, incluant des polices adaptées pour les personnes dyslexiques. Responsive Design : Assurer que le site est utilisable sur divers appareils (ordinateurs, tablettes, smartphones).

CONTRAINTE LÉGALES & RÈGLEMENTAIRES

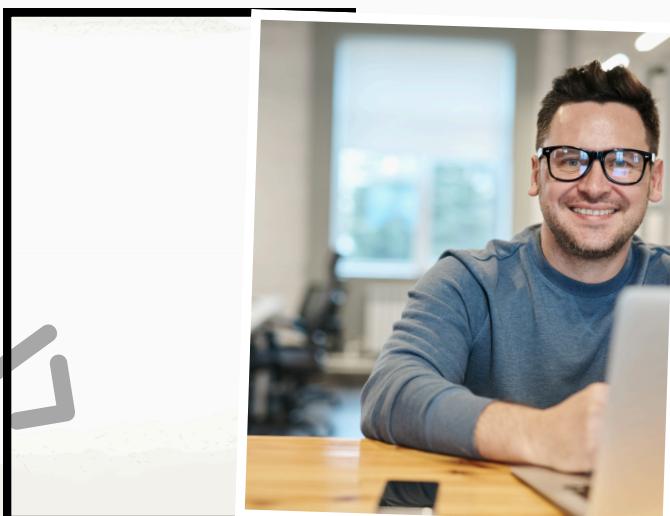
Mentions légales :

La loi pour la confiance dans l'économie numérique (LCEN) continue d'exiger que les sites web fournissent des informations permettant d'identifier l'éditeur du site (nom, prénom, adresse, coordonnées). Cela inclut également les informations sur l'hébergeur. Ces mentions doivent être facilement accessibles pour tout utilisateur visitant le site .

Cookies :

Depuis les lignes directrices actualisées par la CNIL en 2020, il est désormais obligatoire d'obtenir le consentement préalable des utilisateurs avant le dépôt ou la lecture de certains cookies, à l'exception des cookies strictement nécessaires, comme ceux permettant l'authentification ou la gestion des paniers d'achat.

PERSONA



Bio

JULIEN VIENT DE RECEVOIR UNE GÉNÉREUSE DONATION DE 300 000 EUROS DE SES PARENTS POUR L'AIDER À SE LANCER DANS LA VIE. APRÈS AVOIR ÉTUDIÉ DIFFÉRENTES OPTIONS D'INVESTISSEMENT, IL A DÉCIDÉ D'ACHETER UN APPARTEMENT DE 4 CHAMBRES DANS UN QUARTIER PRISÉ DE BORDEAUX.



14/10/1996

Age: 28 ans

Situation: Ingénieur en informatique

Localisation de son bien : Bordeaux



Attentes

- Prix : 650 EUROS par chambres.
- Colocataires : Mixte
- Rentabilité : +1200 €

ARTICLE



Question de M. ROJOUAN Bruno (Allier - Les Républicains-R) publiée le 18/05/2023

M. Bruno Rojouan attire l'attention de M. le ministre délégué auprès du ministre de la transition écologique et de la cohésion des territoires, chargé de la ville et du logement sur les difficultés pour les étudiants étrangers à trouver un logement sans caution française.

Les étudiants étrangers qui souhaitent étudier en France sont souvent confrontés à de nombreuses difficultés, notamment en ce qui concerne la recherche d'un logement. L'une des principales difficultés pour ces étudiants est de trouver un logement sans avoir à fournir une caution française. Selon une étude d'une fédération étudiante (FAGE), près de 40 % des étudiants étrangers rencontrent des difficultés pour trouver un logement en France. Cette situation est particulièrement difficile pour les étudiants qui ne sont pas en mesure de fournir une caution, car cela limite leur choix de logements.

De plus, les propriétaires peuvent être réticents à louer un logement à un étudiant étranger qui ne peut pas fournir une caution française, car cela augmente le risque de non-paiement du loyer ou de dommages au logement. Selon une enquête menée par le site internet SeLoger, les étudiants étrangers sont les moins bien lotis dans le marché locatif en France, car ils doivent souvent faire face à des exigences de garanties supplémentaires. Cela rend la recherche de logement plus difficile et plus longue pour ces étudiants.

Enfin, la forte demande de logements étudiants en France, en particulier dans les grandes villes universitaires, peut rendre la situation encore plus difficile pour ces étudiants. Selon une enquête menée par l'Observatoire de la vie étudiante, le taux d'occupation des résidences universitaires en France est de 93,5 %, ce qui signifie que de nombreux étudiants doivent chercher des logements en dehors des résidences universitaires. Cette concurrence accrue peut rendre la recherche de logement plus difficile pour les étudiants étrangers qui ne peuvent pas fournir une caution française.

<https://WWW.SENAT.FR/question/base/2023/qSEQ23050677.html>
ce lien mène vers l'article.

ARBORESCENCE

SCÉNARIOS POUR LE VISITEUR ET L'UTILISATEUR (PROPRIÉTAIRE, LOCATAIRE)

L'accès aux fonctionnalités du site est différencié selon le statut de l'utilisateur :

UTILISATEUR ENREGISTRÉ :

- Accès complet à l'ensemble de l'arborescence du site
- Peut utiliser la messagerie
- Peut ajouter un appartement (selon son rôle : propriétaire ou locataire)

VISITEUR (NON ENREGISTRÉ) :

- Accès limité au site
- Ne peut pas accéder à la messagerie
- Ne peut pas ajouter d'appartement
- Doit créer un compte pour devenir propriétaire ou locataire et accéder aux fonctionnalités complètes.

SCÉNARIOS POUR L'ADMIN

L'administrateur dispose de privilèges étendus par rapport au propriétaire et au locataire :

GESTION DES APPARTEMENTS :

- Peut supprimer tout appartement ne respectant pas les règles du site.

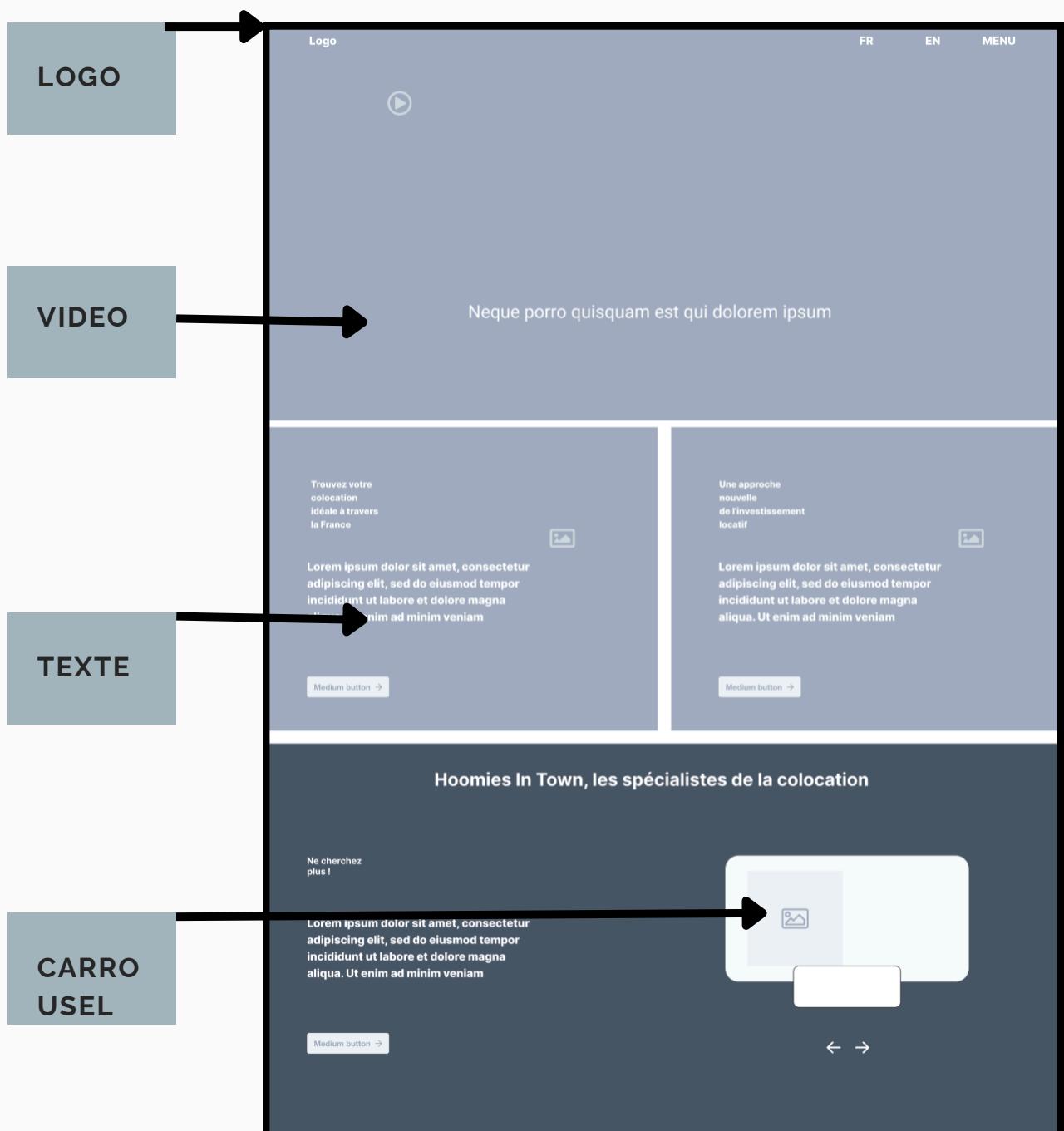
GESTION DES UTILISATEURS :

- A le pouvoir de supprimer les comptes utilisateurs en cas de non-respect des conditions d'utilisation.

MODÉLISATION

Wireframe

VERSION DESKTOP DE LA PAGE D'ACCUEIL





MODÉLISATION

Mockup

VERSION DESKTOP DE LA PAGE COLOCATION

The mockup shows a search interface for finding a room in a shared accommodation. At the top, there's a logo for "Hoomies in Town" and language links for "FR EN". A "Menu" button is also present. The main search bar contains the placeholder "Rechercher une chambre en colocation". Below the search bar are filters for "Ville", "Budget", "Type", and "Nombre de locataires". There are also buttons for "Équipement" and "Réinitialiser la recherche". The results section displays 109 results, showing four preview cards for rooms. To the right, a map of Europe highlights several locations with numbered pins (3, 4, 8) and a red callout bubble showing a price of "380€". At the bottom, there are links for "Mentions légales", "CGU", "Données personnelles et cookies", and a copyright notice: "© 2024 Copyright: Hoomies in Town".

Cette maquette a été vraiment inspirante. On a dû faire quelques ajustements pour rendre les pages plus lisibles, mais elle a posé des bases solides pour notre design final. Les modifications ont permis d'améliorer l'expérience utilisateur tout en gardant l'esprit du concept d'origine.

MODÉLE LOGIQUE DES DONNÉES

RÈGLES DE TRANSFORMATION DU MCD AU MLD

Une entité du MCD se transforme en relation, c'est-à-dire en table.

Une table contient un ensemble d'enregistrements :

1. Une ligne correspond à un enregistrement.
2. Une colonne correspond à un champ.

RÈGLE 1 :

Une table contiendra un ensemble d'enregistrements :

1. Chaque ligne représente un enregistrement individuel.
2. Chaque colonne correspond à un champ de données.

L'identifiant de l'entité devient la clé primaire de la table. La clé primaire permet d'identifier de manière unique chaque enregistrement dans la table. Par conséquent, les valeurs de la clé primaire doivent être uniques et non nulles.

RÈGLE 2 :

Les associations de type 1:N (un-à-plusieurs) .

Lorsqu'une association dans le MCD a une cardinalité maximale de "1" d'un côté et "N" (plusieurs) de l'autre :

1. On crée une clé étrangère dans la table correspondant à l'entité du côté "N".
2. Cette clé étrangère fait référence à la clé primaire de la table correspondant à l'entité du côté "1".

RÉGLE 3 :

Une association de type N:N

Les cardinalités maximales sont "N" des deux côtés se traduit par la création d'une table intermédiaire. La clé primaire de cette table est constituée des clés étrangères des tables des entités associées. Les propriétés de l'association deviennent des attributs de cette table intermédiaire.