Réalisé par :

Mr. BOUZAYANI Malek

Plateforme de gestion des réservations des logements

BOUZAYANI Malek

Table de matières

Listes des figures

Liste des Tableaux

Introduction générale

La location d'un logement pour un court ou long séjour se passe souvent par un intermédiaire, un ami ou un membre de famille. Ceci peut invoquer des difficultés de trouver un endroit qui valide les attentes ou les besoins des locataires. Dans ce contexte s'inscrit notre projet « Plateforme de gestion des réservations des logements ». Il s'agit du développement d'une application mobile permettant aux utilisateurs de consulter, réserver, commenter, évaluer ou signaliser des annonces de location des logements et un site web destiné pour un administrateur permettant la validation des annonces publiées et le contrôle des utilisateurs ou du contenu publié. Le présent rapport s'articule autour de quatre chapitres. Nous commencerons par le chapitre 1 intitulé « Contexte général et étude préalable » qui présente le contexte général de notre projet, l'étude de l'existant et la solution envisagée. Le deuxième chapitre intitulé « Analyse et spécification des besoins » présente la spécification des besoins fonctionnels, non fonctionnels et techniques de notre projet. La modélisation UML est présentée à travers les diagrammes de cas d'utilisation et les diagrammes de séquences système. A la lumière de ce chapitre, nous entamerons le troisième chapitre intitulé « Conception », dans lequel nous présenterons la notation de modélisation utilisée ainsi que l'ensemble des diagrammes conçus. Le quatrième chapitre « Réalisation » est consacré à l'étude technique où nous détaillerons notre environnement de travail suivis d'une présentation des différentes fonctionnalités de notre « Plateforme de gestion des réservations des logements » à travers des captures d'écran. Nous clôturons ce rapport par une conclusion générale sur le travail effectué.

Chapitre 1. Contexte général et étude préalable

Introduction

I- Motivation et objectif du travail

II- Etude de l'existante

III- Solution proposée

IV- Méthodologie de développement

Conclusion

Motivation et objectif du travail

La location d'un logement pour un court ou long séjour se passe souvent par un intermédiaire, un ami ou un membre de famille. Ceci peut invoquer des difficultés de trouver un endroit qui valide les attentes ou les besoins des locataires. C'est dans ce cadre, s'inscrit notre projet de fin d'études qui consiste à concevoir et développer une plateforme pour la gestion des réservations des logements (application mobile et un Dashboard web). À travers la plateforme le client se met en contact direct avec le locateur du logement, il peut visualiser les détails du logement (équipements, localisation, règlements etc. ...), réserver, évaluer, et aussi signaliser les annonces, Ainsi, un locateur à travers la plateforme donne une description détaillée (prix, équipements, intérieur, etc.,) de ces logements aussi confirmer ou décliner les demandes de réservation reçues.

I. Etude de l'existant

L'étude de l'existant constitue une étape préliminaire pour la réalisation d'un projet. Dans cette section, nous menons une analyse détaillée de l'existant, suivie d'une critique et une étude comparative des applications existants, ceci dans l'objectif de dégager les points faibles et les points forts et les besoins des utilisateurs pour les prendre en considération lors de la conception et la réalisation. À la suite nous proposons alors quelques solutions en Tunisie et à l'étranger qui offrent le même concept que notre plateforme.

1.Applications existantes

1.1 Applications en Tunisie

Les sites des annonces de l'immobilité en Tunisie sont nombreux mais peu qui se spécialisent dans le domaine de location, d'où nous citions quelques sites spécialisés en immobilité générale.

1.1.1 Kabylis.tn

Lien : https://www.kabylis.tn

Description : Kabylis est un site de booking en ligne du tourisme alternatif et authentique en Tunisie sous forme d'une Marketplace locale, communautaire, qui réunit les différents acteurs du tourisme authentique et durable pour proposer leurs services aux voyageurs.

Les points forts :

- Présence des évènements ;
- Site réactif.

Les points faibles :

- Absence de plate-forme mobile ;
- Absence d'un module de discussion.

La figure 3 présente l'interface d'accueil du site Kabylis.tn

Figure 1 kabylis.tn

1.1.2 Mubawab.tn

Lien : https://www.mubawab.tn

Description : mubawab.tn est un site web crée en 2012 au Maroc en 2019 Mubawab a fait l'acquisition de Jumia House dans la région de Maghreb d'où il ouvre ses portes à l'Algérie et la Tunisie.

Les points forts :

- Version mobile ;
- Plate-forme multinationale.

Les points faibles :

- Structuration de zone de recherche ;
- Absence de module de réservation et discussion.

La figure 4 présente l'interface d'accueil du site Mubawab.tn

Figure 2 Mubawab.tn

La figure 5 présente l'interface de l'application Mubawab.tn

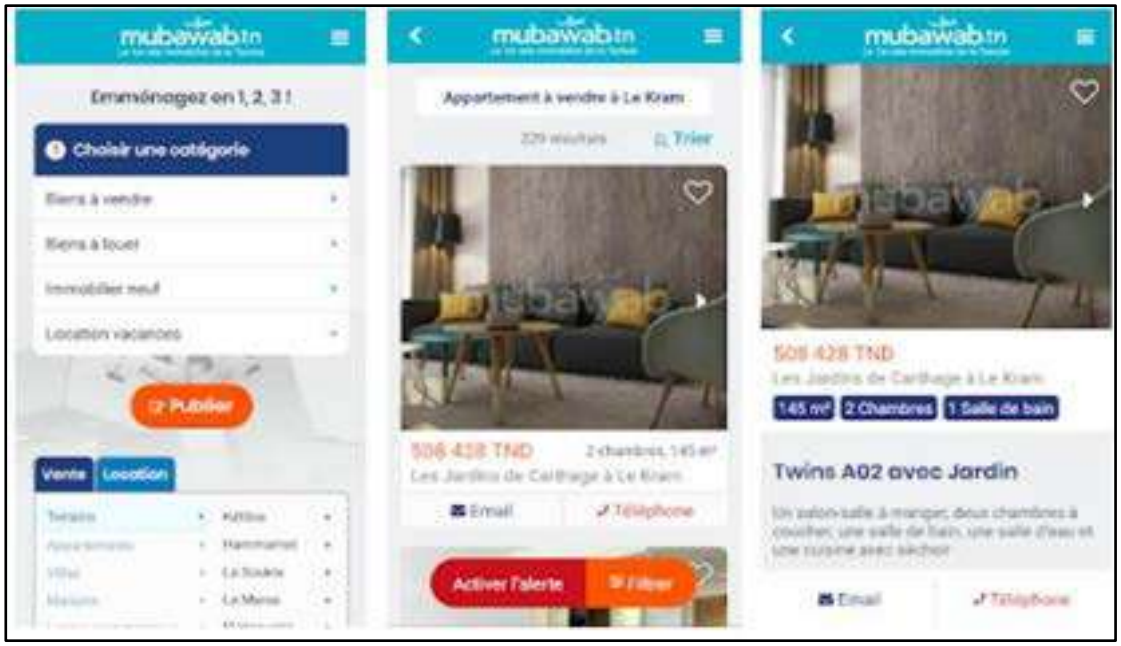

Figure 3 Mubawab Mobile

1.1.3 Bnb.tn

Lien : https://www.bnb.tn

Description : BnB Tunisie est un site d'annonces immobilières en Tunisie permettant aux annonceurs (agences immobilières, intermédiaires, particuliers) de publier leurs offres de vente, location ou location de vacances. BnB Tunisie est une plateforme qui regroupe les agences immobilières agréées et les particuliers en un seul endroit.

Les points forts : Version mobile

Les Points faibles :

- Absence de module de réservations et discussion ;
- Application mobile n'a pour rôle qu'à visualiser les annonces (toute opération nécessite l'ouverture du portail web)

La figure 6 présente l'interface d'accueil du site Bnb.tn

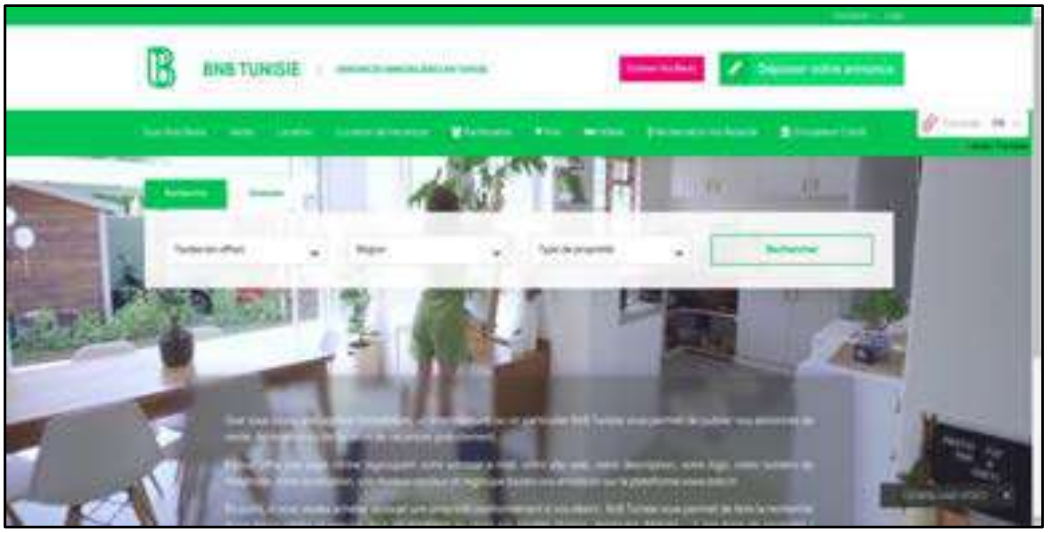

Figure 4 BnB.tn

La figure 7 présente l'interface de l'application Bnb.tn :

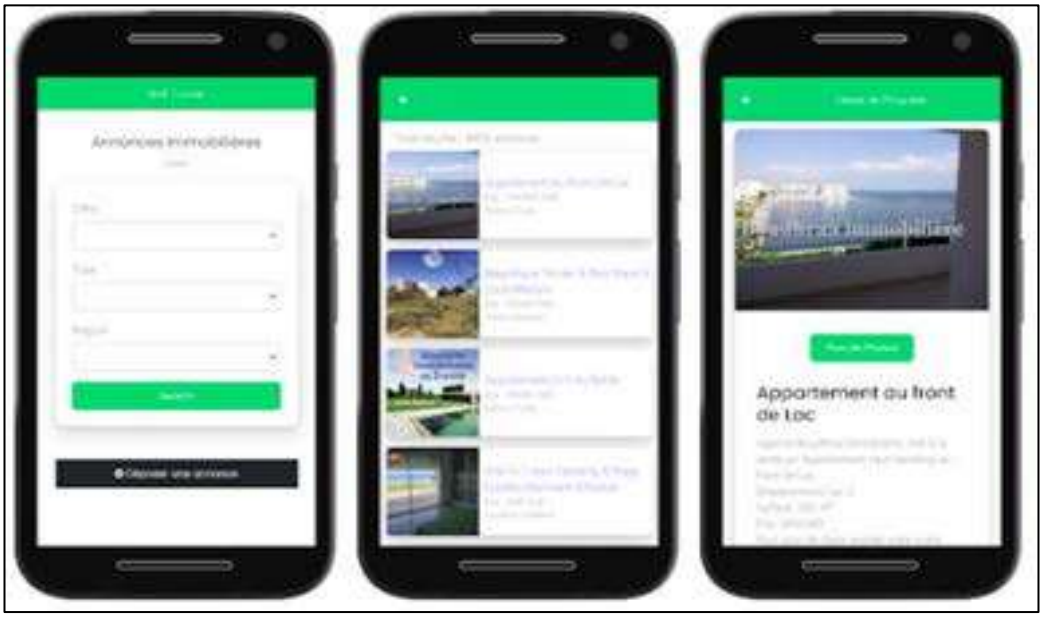

Figure 5 BnB Mobile

1.2 Application à l'étranger

1.2.1 Airbnb.com

Site : https://www.airbnb,com

Description : Créée en 2007, alors que 2 hôtes accueillaient 3 voyageurs dans leur logement de San Francisco, la communauté Airbnb compte désormais 4 millions d'hôtes, qui ont fait vivre plus de 900 millions de séjours à des voyageurs dans presque tous les pays du monde.

Les points forts :

- Paiement en ligne
- Innovant et très dynamique
- C ouverture internationale
- Application mobile

Les points faibles :

- Frais des services élevés
- Paiement qu'en devise

La figure 8 présente l'interface d'accueil du site Airbnb.com

Figure 6 Airbnb.com

La figure 9 présente l'interface de l'application Airbnb

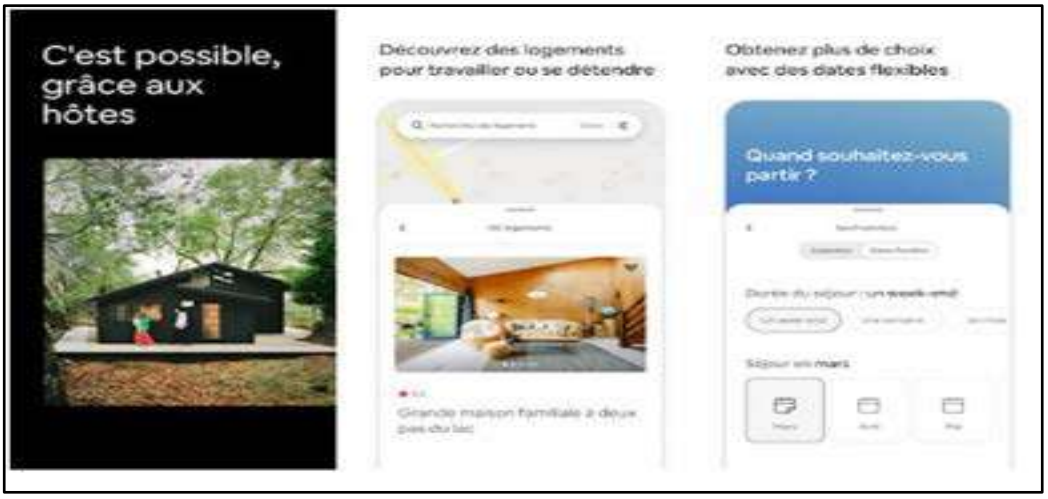

Figure 7 Airbnb Android

2.Critique de l'existant

Nous pouvons classer le résultat de l'analyse des applications existantes mentionnées précédemment selon quelques critères [Cx] pris en considération dans le processus d'évaluation de ses applications :

- [C1] Interface utilisateur et Attirance : la qualité du contenu et l'exhaustivité ;
- [C2] Espace Clients : la possibilité de s'inscrire ;
- [C3] Existence d'une version mobile : possibilité d'installer une application sur un smartphone ;
- [C4] Méthode de discussion : donner des avis, échanger des idées, commenter ;
- [C5] Géolocalisation : disponibilité d'une carte de guidage ;
- [C6] Temps de réponse : la rapidité de chargement des informations.

Le tableau 1 présente une comparaison fonctionnelle entre les solutions existantes étudiées

Application	C1	C2	C3	C4	C5	C6
K abylis.tn	4	oui	non	oui	oui	1 062 ms
Bnb.tn	2	oui	oui	non	non	591 ms
Mubawab.tn	3	oui	oui	non	oui	565 ms
Airbnb.com	5	oui	oui	oui	oui	426 ms

Tableau 1 Tableau Comparatif des applications web existantes

II. Solution proposée

Tenant compte des critiques de l'existant et des résultats mentionnés précédemment, il s'avère qu'aucune solution existante ne répond convenablement à notre problématique. Nous sommes amenés alors à proposer une solution digne de confiance et qui pourra répondre à nos besoins, nos objectifs. La solution proposée doit pallier les lacunes constatées dans l'étude des applications existantes.

Notre solution proposée consiste de concevoir et réaliser une application qui soutient le système de gestion des annonces de location des logements et vise à assurer les fonctionnalités suivantes :

- Accès libre aux annonces postulés par des locateurs ;
- La favorisation des annonces (Liste favoris) ;
- La communication (discussion) ;
- La réservation ;
- La gestion des annonces ;
- La réclamation des annonces ou des commentaires qui ne respectent pas l'ordre général ;
- Évaluation et ajout de commentaires sur les annonces par les clients ;
- Un tableau de bord permettant l'analyse des données, le traitement des réclamations et la validation des annonces par l'administrateur.

III. Méthodologie de développement

Une méthodologie de développement est un cadre utilisé pour structurer, planifier et contrôler le développement d'une application. C'est le cadre employé pour modéliser un système avant de sa réalisation pour bien comprendre son fonctionnement et assurer sa cohérence. Suivre une méthodologie de développement permet de baser un projet sur un modèle

de conception et des formalismes de notations ce qui implique la réduction des coûts et des délais.

Il existe une diversité de méthodologies disponibles et chacune offre des avantages.

1.Étude de méthodologie de développement

Devant nombreuses méthodes disponibles, le choix parmi elles devient difficile. Nous étudions à ce propos quelques méthodes de développement pour dégager la méthodologie la plus adéquate à notre projet.

1.1 Agile

Agile est une méthodologie qui anticipe le besoin de flexibilité et applique un niveau de pragmatisme dans la livraison du produit fi ni. Agile nécessite un changement de culture dans de nombreuses entreprises, il se concentre sur la livraison propre de pièces individuelles ou de parties du logiciel et non sur l'ensemble de l'application. Les méthodes agiles contiennent 4 valeurs fondamentales :

- Individus et interactions sur les processus et les outils ;
- Logiciel de travail sur une documentation complète ;
- Collaboration des clients à la négociation de contrats ;
- Répondre au changement au sujet d'un plan.

La figure 9 représente les étapes du processus agile.

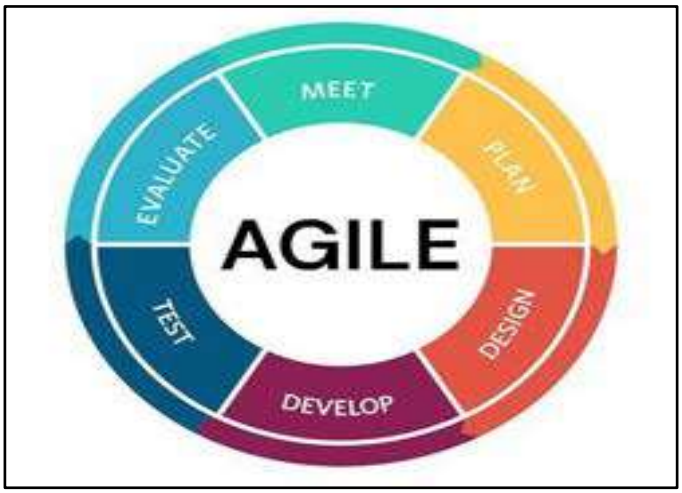

Figure 8 Les étapes du processus Agile

1.2 Scrum

Dérivé d'une méthode de gestion de projets agile, la méthode SCRUM définit un cadre de travail permettant la réalisation des projets complexes. Initialement prévu pour le développement de projet type « Software », cette méthode peut être appliquée à tout type de projet, du plus simple au plus innovant, et d'une manière très facile. Cette méthodologie permet de s'adapter rapidement aux changements d'un client à fréquence régulière. Chaque fin d'itération, l'équipe et le client réévaluent les spécifications du logiciel.

Le cycle de vie d'un projet Scrum, est divisé en trois parties :

- Phase d'initialisation : phase linéaire aux processus explicitement connus et dont les inputs et les outputs sont bien déterminés.
- 'Sprint' de développement : est un processus empirique.
- Phase de clôture : est aussi une phase linéaire dont les processus sont définis clairement.

La figure 11 représente les étapes du processus Scrum.

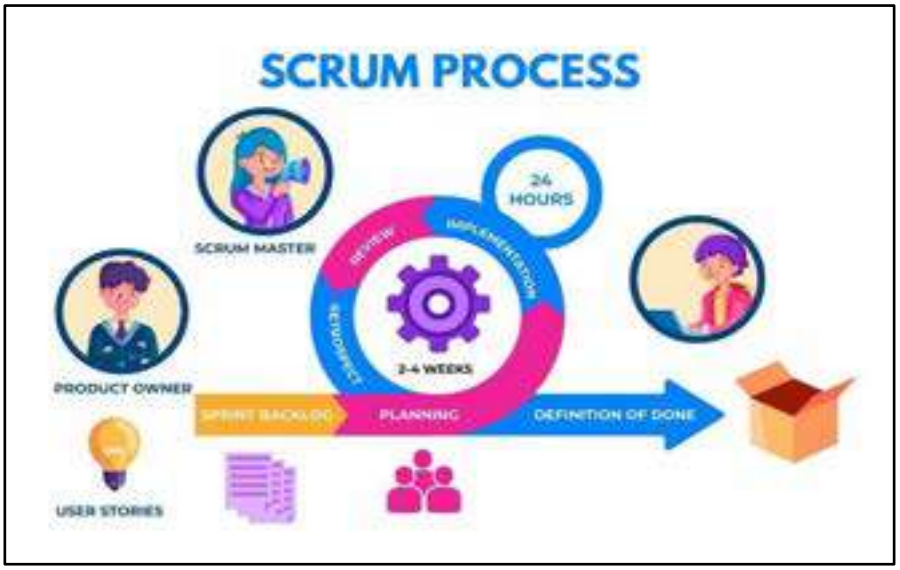

Figure 9 Les étapes du processus Scrum

1.3 Unified Process

Unified Process est un processus de développement centré sur l'architecture, basé sur des cas d'utilisation, itératif et incrémental qui utilise le langage de modélisation unifié (UML). Unified

Process peut être appliqué à différents systèmes logiciels avec différents niveaux de complexité technique et de gestion dans différents domaines et cultures organisationnelles.

La figure 12 représente les étapes du processus Unified Process.

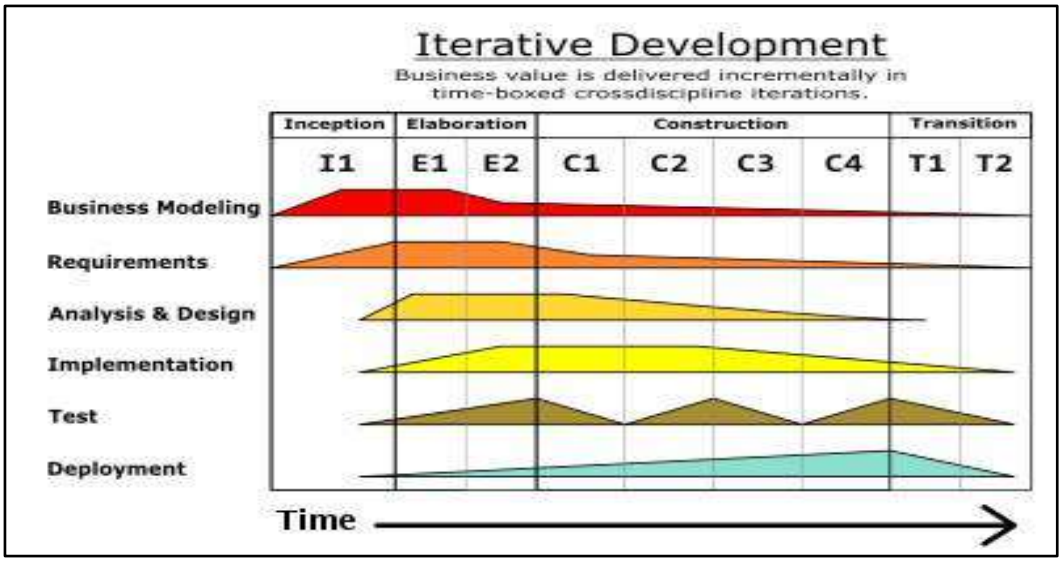

Figure 10 Les étapes du processus Unified Process

1.3.1 Two Tracks Unified Process (2TUP)

Two Tracks Unified Process (2TUP) est un processus unifié. Le principe de 2TUP est que toute évolution imposée à un logiciel peut être décomposée et traitée en parallèle, selon un axe fonctionnel et un axe technique, et la réalisation du logiciel consiste à fusionner les résultats de ces deux branches du processus.

1.3.2 Rational Unified Process (RUP)

C'est un processus de développement des logiciels de Rational, une division d'IBM. Il divise le processus de développement en quatre phases :

- Interception : l'idée de ce projet est déclarée. L'équipe de développement détermine si le projet mérite d'être poursuivi et quelles ressources seront nécessaires.
- Élaboration : l'architecture du projet et les ressources nécessaires sont encore évaluées. Les développeurs considèrent les applications possibles du logiciel et les coûts associés au développement.
- Construction : le projet est développé et complété. Le logiciel est conçu, écrit et testé.

- Transition : le logiciel est diffusé au public. Les derniers ajustements ou mise à jour sont effectués en fonction des commentaires finaux des utilisateurs.

2.Étude comparative et choix

Méthodologies	Avantages	Inconvénients
Agile	Agile offre de multiples opportunités pour l'engagement des parties prenantes et des équipes. En impliquant le client dans chaque étape du projet, il y a un haut degré de collaboration entre le client et l'équipe du projet, ce qui permet à l'équipe de mieux comprendre la vision du client.	- Il a moins bien réussi à intégrer les tests et les opérations à ce mélange. - Manque d'emphase sur la technologie, ce qui peut rendre difficile la vente du concept aux cadres supérieurs qui ne comprennent pas le rôle que la culture joue dans le développement de logiciels.
Scrum	- Le client est au cœur du Projet. - Esprit d'équipe. - Communication lâchée. - Simplicité, efficacité et qualité. - Flexibilité au changement. - Avancement basé sur le concret.	- Manque de pratique pour obtenir le 'DONE' dans le contexte logiciel
Unified Process	- Itératif et incrémental. - Focalisé sur les risques. - Des approches éprouvées pour développer et maintenir des logiciels de qualité. - Meilleur niveau de portabilité	- Nécessite des experts - Coûteux à personnaliser - Très axé processus
RUP	- Traçabilité à partir des cas d'utilisation jusqu'au déploiement. - Approche basée sur l'architecture. - Gestion des risques dans les Projets. - Cadre propice à la réutilisation.	- Coût de personnalisation souvent élevé. - Processus très axé. - Vision non évidente ni immédiate.
2TUP	- Convenable pour tout type de projet. - Permet de répondre aux besoins des utilisateurs rapidement. - Canalise et modélise toutes les	Pas de documents types.

	étapes du développement d'un logiciel.	

Tableau 2 Comparatif des méthodologies de développement

3. Choix de la méthodologie

Notre projet est basé sur un processus de développement bien défini qui va de la détermination des besoins fonctionnels attendus du système jusqu'à la conception et le codage final. C'est pour cela qu'on a besoin d'un cycle de développement qui dissocie les aspects techniques des aspects fonctionnels tout en commençant par une étude préliminaire. Ainsi, suite à une étude comparative, notre choix s'est alors porté vers la méthode 2TUP vu qu'elle est caractérisée par une approche nouvelle et originale et qu'elle respecte le cadre de notre projet.

3.1 Mise à pratique du processus 2TUP

2TUP est un processus unifié (c'est-à-dire construit sur UML, itératif, centré sur l'architecture conduit par les cas d'utilisation).

La figure 13 décrit les étapes du processus 2TUP.

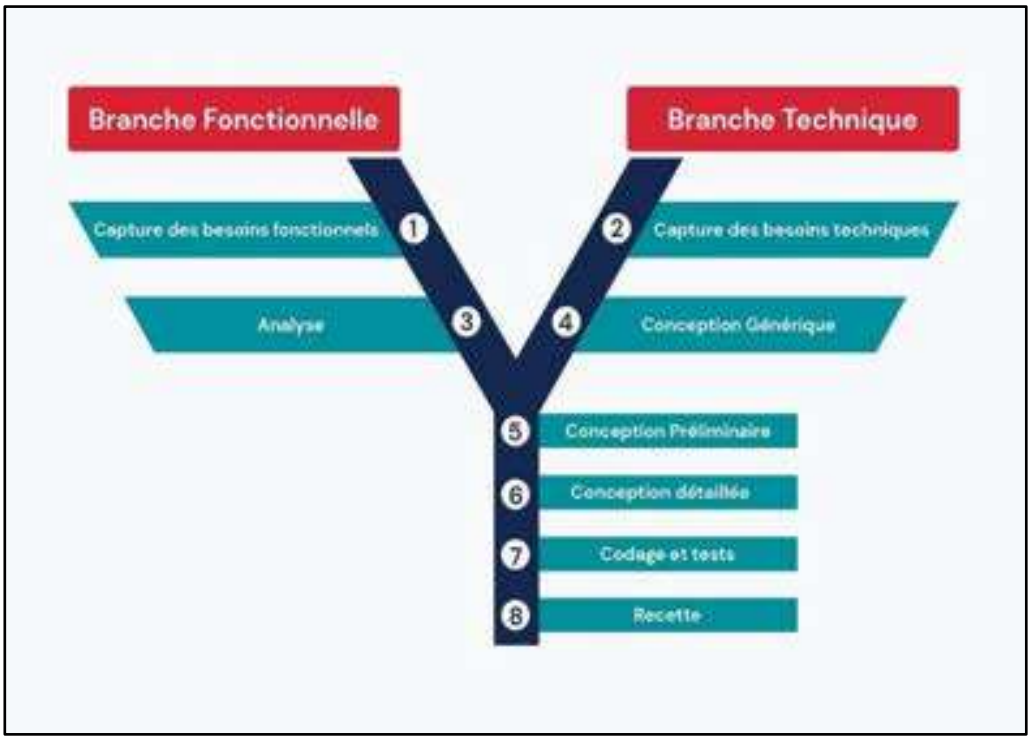

Figure 11 Les étapes du processus 2TUP

Le principe fondateur du 2TUP est que toute évolution imposée à un logiciel peut se décomposer et se traiter parallèlement, suivant un axe fonctionnel et un axe technique. La réalisation du logiciel consiste à fusionner les résultats de ces deux branches du processus. Selon le processus 2TUP que nous suivons, nous passons obligatoirement par les phases suivantes à la suite de notre travail :

- L'étude préliminaire : qui contient une description du service et les cas d'utilisation principaux, c'est une première version de la spécification générale.
- La capture des besoins fonctionnels : qui définit le quoi faire à travers une spécification générale qui décrit le service à développer d'un point de vue fonctionnel et une spécification détaillée qui précise les traitements qui concernent chaque scénario des cas d'utilisation présent en spécification générale tout en respectant les contraintes fonctionnelles et non fonctionnelles.
- L'analyse : où on effectue simultanément l'étude des données et l'étude des traitements à effectuer.
- La capture des besoins techniques : qui permettent de satisfaire les contraintes techniques présentes dans le cahier des charges et donc répondre aux attentes de client.
- La conception générique : qui définit le comment faire.
- La conception détaillée : qui précise l'implémentation technique de l'application. Elle consiste en la fusion de la spécification détaillée et de la conception générique, pour déterminer comment faire le quoi faire dans le détail. On y trouve le schéma de base de données, les diagrammes de classes et les diagrammes des séquences supplémentaires qui détaillent les interactions entre les composants du logiciel.
- Le Codage et les Tests : qui décrivent comment est l'application est réalisée et installée sur un environnement d'exécution et comment la stratégie de validation est effectuée.

Conclusion

Dans ce premier chapitre, nous avons présenté le contexte général du projet et nous avons étudié les solutions existantes. Nous avons ensuite comparé les différentes méthodes de développement et nous avons présenté la méthodologie 2TUP que nous allons adoptée pour le développement de notre projet.

Chapitre 2. Analyse et spécification des besoins

Introduction

I- Besoins fonctionnels et non fonctionnels

II- Analyse semi-formelles des besoins

Conclusion

Introduction

Le choix et la mise en pratique de la méthodologie 2TUP nous mènent au présent chapitre qui sera consacré à l'identification de toutes les fonctionnalités de notre futur système et ceci en recensant les besoins fonctionnels et appréhendant la liste des exigences traduites par les besoins non fonctionnels. Les besoins techniques doivent également être recensés pour garantir une description sans ambiguïté du projet à développer.

I. Besoins fonctionnels et non fonctionnels

1.Identification des acteurs

Tout système interactif, doit assurer et faciliter l'interaction avec ces utilisateursfinaux appelés aussi « les acteurs ». Un acteur représente le rôle d'une entité externe exploitant le système à travers ses différentes interfaces.

Par la suite, nous avons détaillé les acteurs et rôles de notre plateforme de réservation des logements :

- Internaute : C'est un visiteur de l'application, il peut consulter les annonces, comme il peut effectuer des recherches. Il a aussi le droit de créer un profil et devenir ainsi un utilisateur.
- Utilisateur : C'est un internaute qui possède un compte dans l'application. Il peut Commenter, réserver et reporter des annonces ou des commentaires. Il a le droit depublier des annonces ou de faire des réservations,
- Manager : C'est un utilisateur qui à publier une ou plusieurs annonces de logements. Il peut modifier ou supprimer ces annonces ainsi que valider ou supprimer des réservations desces annonces
- Administrateur : C'est le modérateur du système il gère la validation des annonces, et consulte aussi les reports crées par les utilisateurs.

2. Identification des besoins

Dans la suite, nous allons exposer les besoins fonctionnels de notre application de réservation des logements en étroite relation avec les acteurs précédemment mentionnés.Nous rappelons que ces besoins doivent répondre aux exigences du futur système.

2.1 Les besoins fonctionnels

2.1.1 Les besoins fonctionnels de l'acteur « Internaute »

L'internaute (Visiteur) détient le droit de :

- Consulter les annonces : il peut consulter toutes les annonces,
- Effectuer des recherches : il peut effectuer des recherches par localisation ou par les autres critères disponibles,
- Créer un compte : il peut créer un compte

2.1.2 Les besoins fonctionnels de l'acteur « Utilisateur » :

L'utilisateur détient le droit de :

- S'authentifier : il doit saisir son adresse E-mail et son mot de passe pour accéder son profil ;
- Mettre à jour ces informations : il peut modifier et mettre à jour ses propres informations ;
- Passer une réservation : il peut passer une réservation en spécifiant une durée ;
- Contacter les locateurs : il peut envoyer un message aux propriétaires des annonces ;
- Commenter et évaluer une annonce : il peut commenter et évaluer les annonces publiées ;
- Gérer les Favoris : il peut enregistrer des annonces qui l'intéressent dans sa liste de favoris ou les supprimer ;

- Envoyer une réclamation : il peut envoyer une réclamation d'une annonce ou d'un commentaire à l'administrateur via un formulaire dans lequel il explique sa demande ;
- Publier une annonce : il peut publier une annonce.

2.1.3 Les besoins fonctionnels de l'acteur « Manager» :

Le Manager détient le droit de :

- Gérer ses annonces : il peut consulter, modifier ou supprimer ces anciennes annonces.
- Gérer ces réservations : il peut consulter les demandes de réservation de cesannonces et il peut confirmer ou décliner.

2.1.4 Les besoins fonctionnels de l'acteur «Administrateur» :

L'administrateur détient le droit de :

- Gérer les annonces publiées : il vérifie toutes les annonces publiées. Il peut soit les
- valider, soit les supprimer.
- Gérer les comptes des abonnés : il a un accès à la liste des utilisateurs. Il à le droit deles supprimer ou les bloquer.
- Traiter les réclamations : il consulte les réclamations des utilisateurs. Il peut supprimer l'annonce ou bien bloquer l'abonné qui a lancé la signalisation.

2.2 Besoins non fonctionnels

Un besoin non fonctionnel est une restriction ou une contrainte qui pèse sur un service du système, telle les contraintes liées à l'environnement et à l'implémentation et les exigences en matière de performances.

Le tableau 3 décrit les besoins non fonctionnels souhaitée.

Besoins non fonctionnels	Description
 **Confidentialité**	L'authentification est obligatoire afin de bénéficier de quelques services.
 **Ergonomie**	Tous les standards d'ergonomie doivent être respectés dont la convivialité et la compréhensibilité des interfaces graphiques. De plus, le choix des couleurs, la densité et l'organisation des éléments sur l'écran et

	aussi l'utilisation des messages informatifs et des messages d'erreurs bien formées et bien lisibles.
Disponibilité et Maintenabilité	L'application doit être accessible à tout moment et devra être extensible, c'est-à-dire facilement maintenable et s'adapte aux nouvelles exigences en cas de modification ou d'ajout d'une fonctionnalité.
Performance	Nous souhaitons réaliser une application mobile qui offre un temps de chargement acceptable par l'utilisateur et résulte une expérience utilisateur fluide en ce qui concerne la navigation dans l'application.
Sécurité	Les contraintes sécuritaires à prendre en compte lors de la réalisation de notre application sont l'attribution des rôles à chaque groupe d'utilisateurs pour s'assurer que l'accès aux différents espaces est protégé.
Simplicité	Notre application doit être simple à utilisateur et ne nécessite pas des connaissances poussées puisqu'elle s'adresse à un public de différent âge incluant des non informaticiens.

Tableau 3 Besoins non fonctionnels

II. Analyse semi-formelles des besoins

Dans la cadre de notre projet, nous avons opté pour le langage UML comme un langage de conception.

UML est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système.

1.Diagramme de cas d'utilisation

1.1 Cas d'utilisation général

La figure 14 illustre le diagramme de cas d'utilisation global qui permet de définir les principales interactions entre les acteurs et les besoins fonctionnels afin d'avoir une vue globale sur notre système.

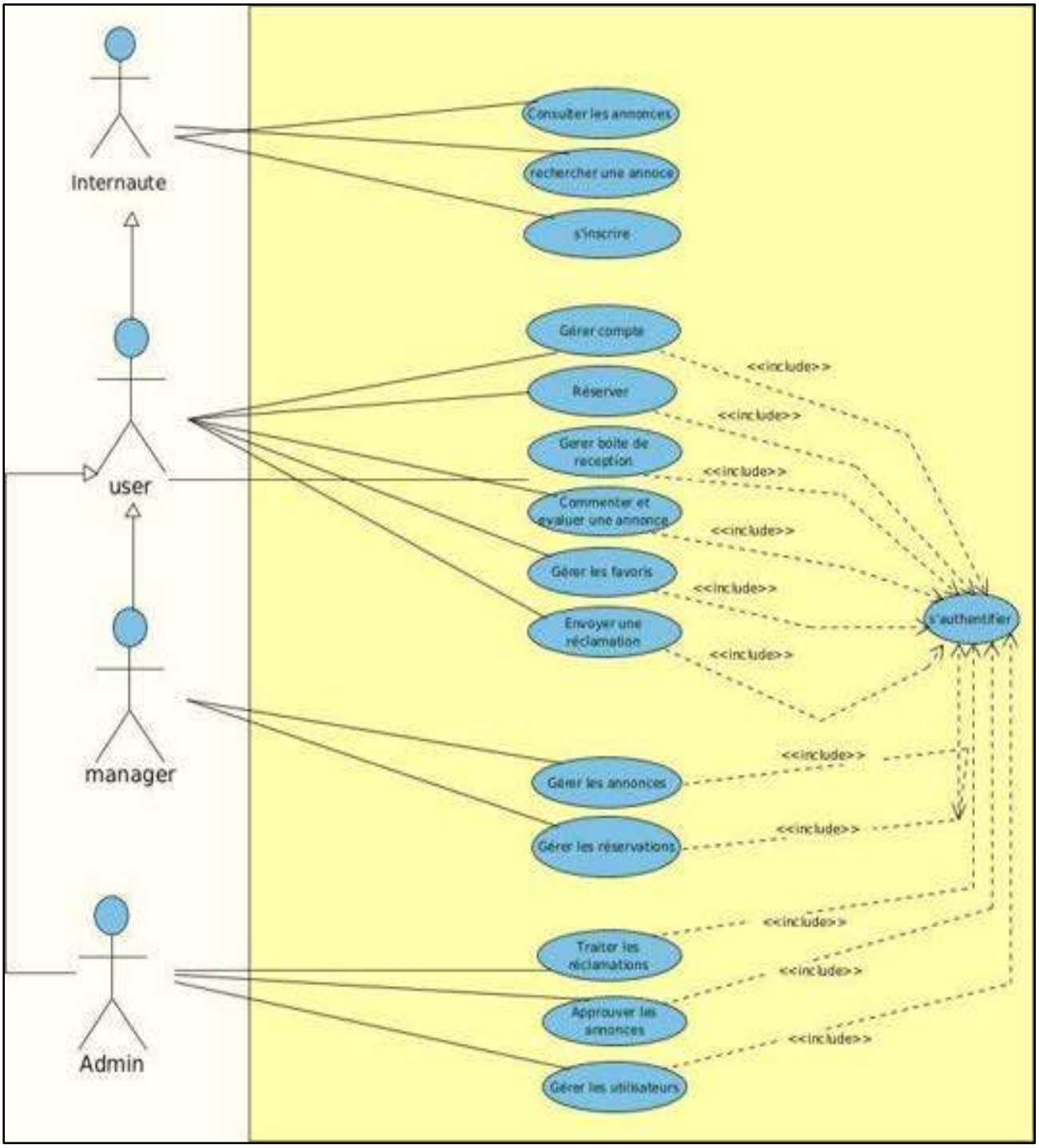

Figure 12 Diagramme de cas d'utilisation général

Le tableau 4 représente les différents cas d'utilisation, les acteurs impliqués et une description pour chaque cas.

Cas d'utilisation	Acteur(s)	Description
Consulter les annonces	Internaute	L'internaute peut consulter les annonces publiées
Rechercher un logement	Internaute	Il peut rechercher des annonces selon des choix.
S'inscrire	Internaute	Il peut créer un profil dans l'application pour devenir un Utilisateur.
Gérer Compte	Utilisateur	L'utilisateur peut mettre à jour ces informations
Réserver	Utilisateur	L'utilisateur peut réserver un séjour dans un logement
Gérer boite de réception	Utilisateur	L'utilisateur peut échanger des messages avec les Managers
Commenter et évaluer une annonce	Utilisateur	Les utilisateurs peuvent commenter et évaluer lesannonces.
Gérer les favoris	Utilisateur	L'utilisateur peut enregistrer des annonces qui l'intéressent dansses favoris ou les supprimer.
Envoyer une réclamation	Utilisateur	L'utilisateur peut envoyer une réclamation à l'administrateur via un formulaire dans lequel il explique sa demande.

Gérer les annonces	Manager	Le manger peut consulter, modifier ou supprimer ces anciennes annonces.
Gérer les réservations	Manager	Le manager peut consulter les demandes de réservation de ces annonces et il peut confirmer ou décliner.
Traiter la réclamation	Administrateur	L'administrateur consulte les réclamations des utilisateurs. Il peut supprimer l'annonce ou bien bloquer l'utilisateur qui a lancé la signalisation
Approuver les annonces	Administrateur	L'administrateur vérifie toutes les annonces publiées. Il peut soit les valider, soit les supprimer.
Gérer les utilisateurs	Administrateur	L'administrateur à un accès à la liste des utilisateurs de l'application. Il a le droit de lessupprimer

Tableau 4 Cas d'utilisation

1.2 Cas d'utilisation associés à « Internaute »

La figure 16 illustre le diagramme de cas d'utilisation associés à l'internaute.

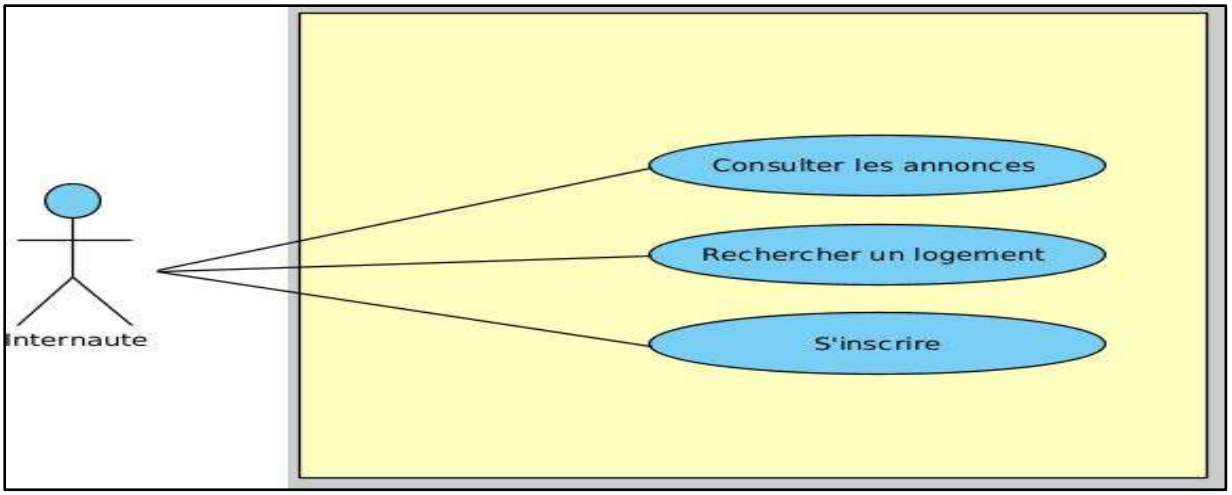

Figure 13 Cas d'utilisation associés à « Internaute »

1.3 Cas d'utilisation associés au « Utilisateur »

Après l'authentification, l'utilisateur est autorisé à faire les fonctionnalités illustrées par laFigure ci-dessous. Nous citons : Consulter les annonces, les commenter, évaluer ou encore il peut gérer sa boîte messagerie.

La figure 17 illustre le diagramme de cas d'utilisation associés à l'utilisateur.

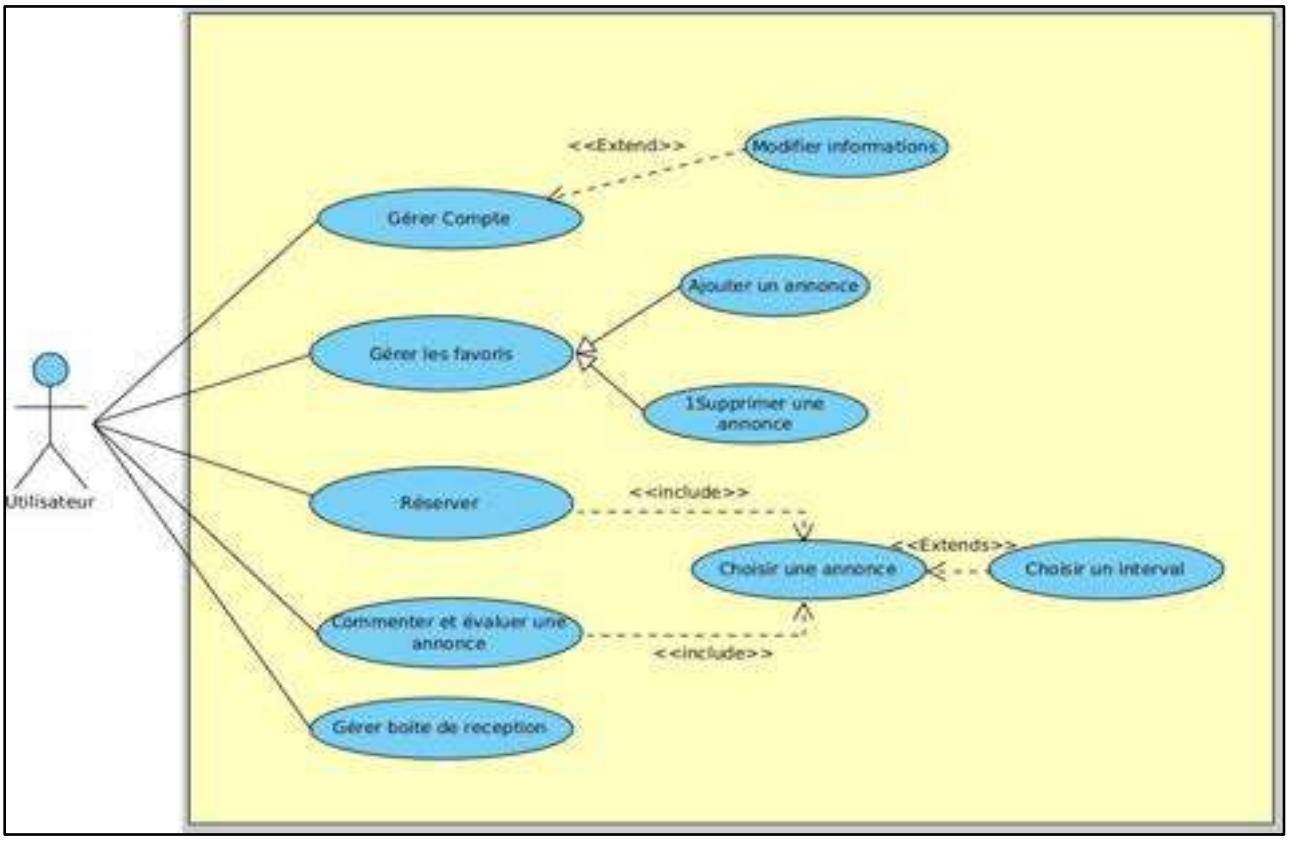

Figure 14 Cas d'utilisation associés au « Utilisateur »

1.4 Cas d'utilisation associés au « Manager »

Après l'authentification, le manager est autorisé à faire les fonctionnalités illustrées parLa figure 18.

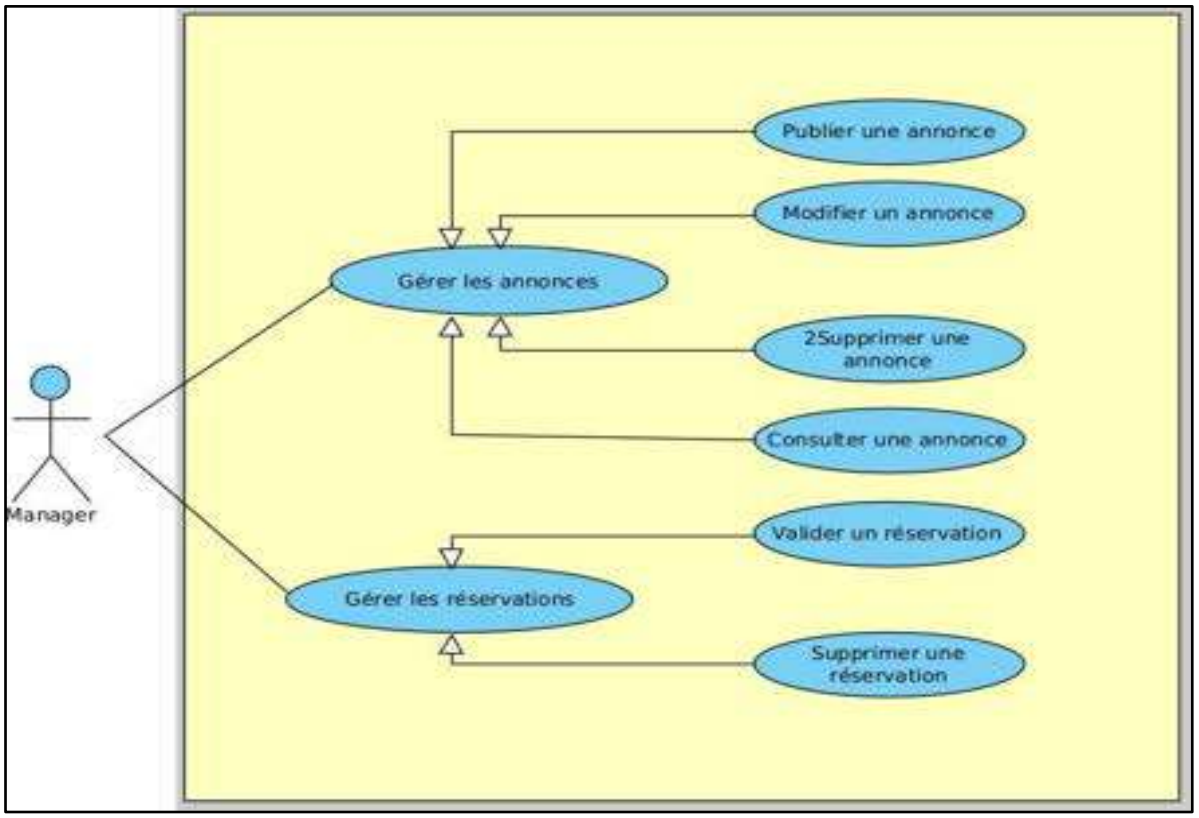

Figure 15 Cas d'utilisation associés au « Manager »

1.5 Cas d'utilisation associés à l'Administrateur

L'administrateur, après l'authentification, il peut assurer les fonctionnalités illustrées parla figure 19. Il peut gérer la liste des utilisateurs de l'application et les annonces publiées.Comme il peut traiter les réclamations des utilisateurs et valider les annonces.

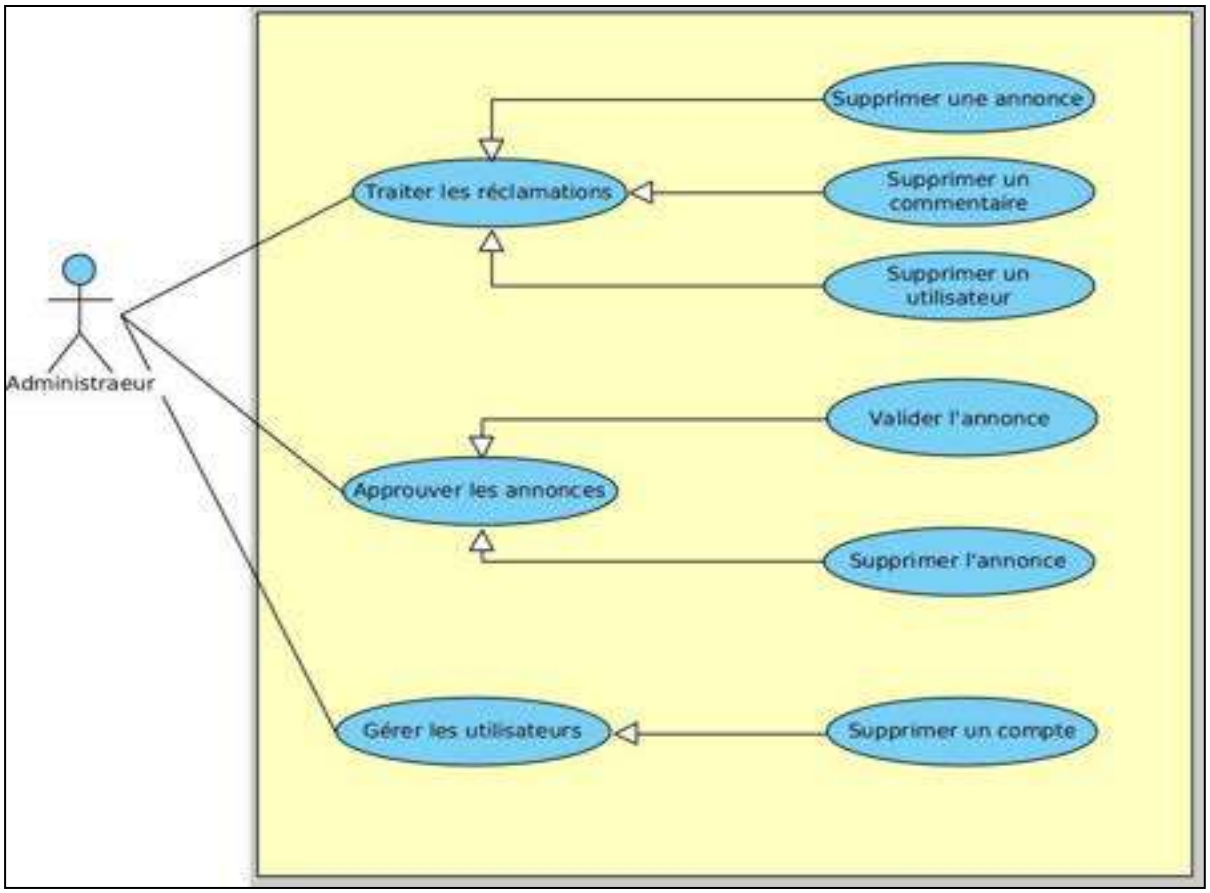

Figure 16 Cas d'utilisation associés à l'Administrateur

2.Diagrammes de séquence système

Le diagramme de séquence est la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Ce diagramme est utilisé pour représenter certains aspects dynamiques d'un système : dans le contexte d'une opération, d'un système, d'un sous-système, d'un cas d'utilisation (un scénario d'un cas d'utilisation) selon un point de vue temporel. Dans ce qui suit, une représentation de quelques scénarios.

2.1 Diagramme de séquence « S'inscrire »

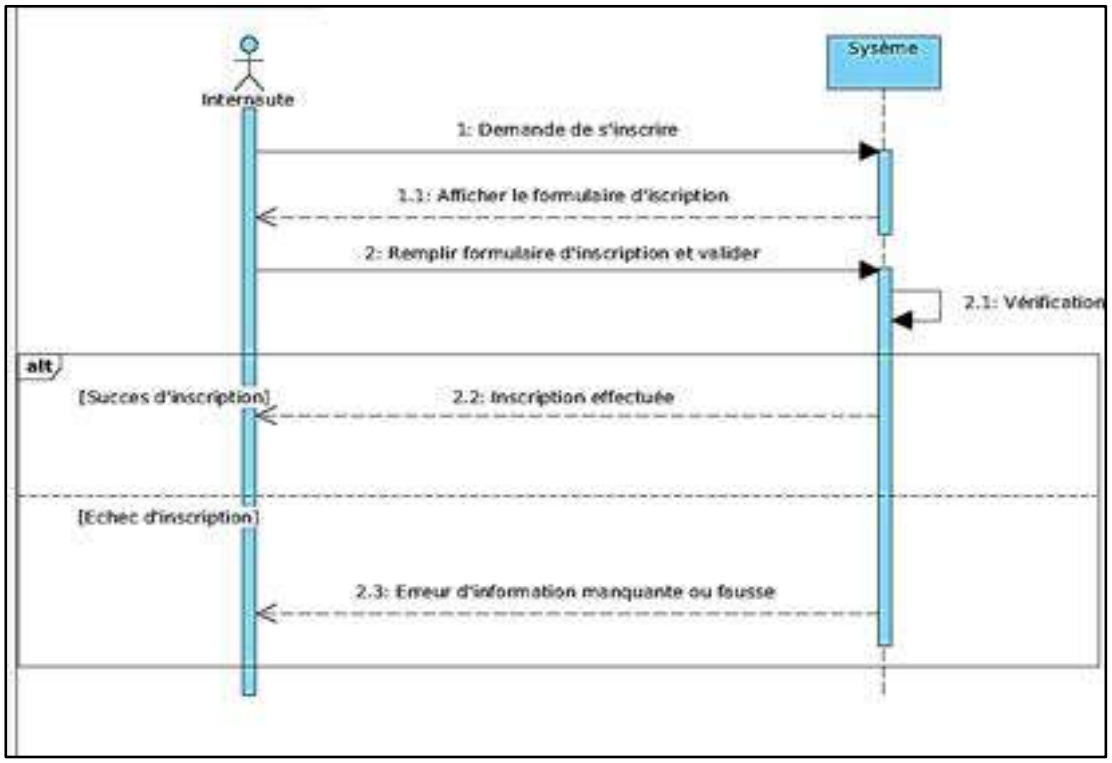

Figure 17 Diagramme de séquence « S'inscrire »

Cas d'utilisation	S'inscrire
Acteur	Internaute
Intérêts	Il permet à l'internaute de créer un profil pour devenir un utilisateur.
Pré condition	Accéder à la page d'inscription
Scénario nominal	1. L'internaute demande l'inscription à l'application. 2. Le système affiche le formulaired'inscription. 3. L'internaute remplit le formulaire et ilvalide. 4. Le système effectue l'inscription.
Post condition	Un nouvel abonné est inscrit.
Scénario alternatif	3.1. Le système indique à l'internaute qu'il y à des champs vides. 3.2. L'internaute remplit les champs

	vides etil valide. - Reprise de l'étape 4 du scénario nominal.
Exception	Utilisateur non inscrit

Tableau 5 Description textuelle du diagramme de séquence « S'inscrire »

2.2 Diagramme de séquence « Authentification »

Le tableau n°6 et la Figure n° 21 ci-dessous décrivent le diagramme de sequence «Authentification».

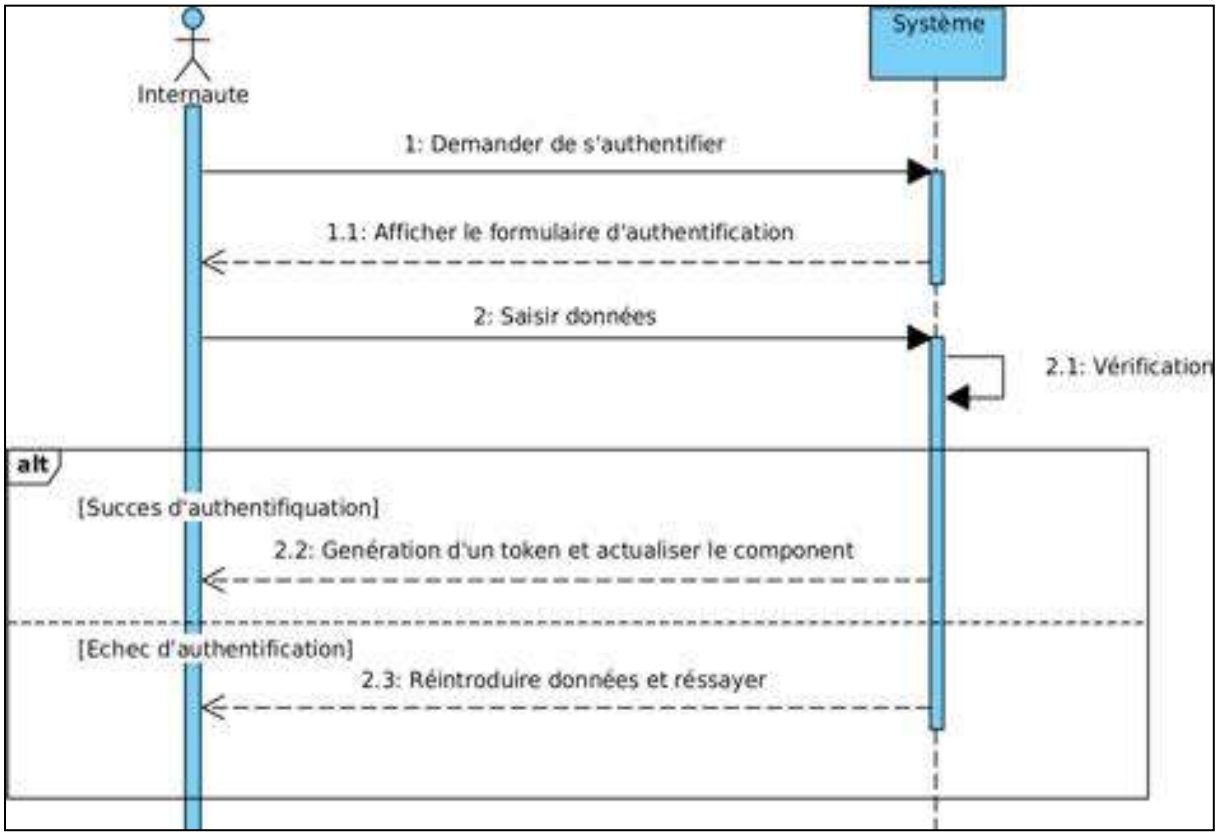

Figure 18 Diagramme de séquence « Authentification »

Cas d'utilisation	Authentification
Acteur	Internaute
Intérêts	Il permet à l'internaute de s'authentifier.
Pré condition	Accéder à la page d'authentification.
Scénario nominal	1. L'internaute demande de s'authentifier. 2. Le système affiche le formulaire d'authentification. 3. L'internaute saisie les données. 4. Le système vérifie les données saisies. 5. Génération d'un token d'accès et mise àjour de l'écran
Post condition	L'internaute est authentifié.
Scénario alternatif	4.1 Le système indique que l'utilisateur non autorisé.
Exception	Utilisateur non authentifié.

Tableau 6 Description textuelle du diagramme de séquence « Authentification »

2.3 Diagramme de séquence « Modifier annonces »

Le tableau n° 7 et la figure n°22 décrivent le diagramme de séquence « Modifier annonce »

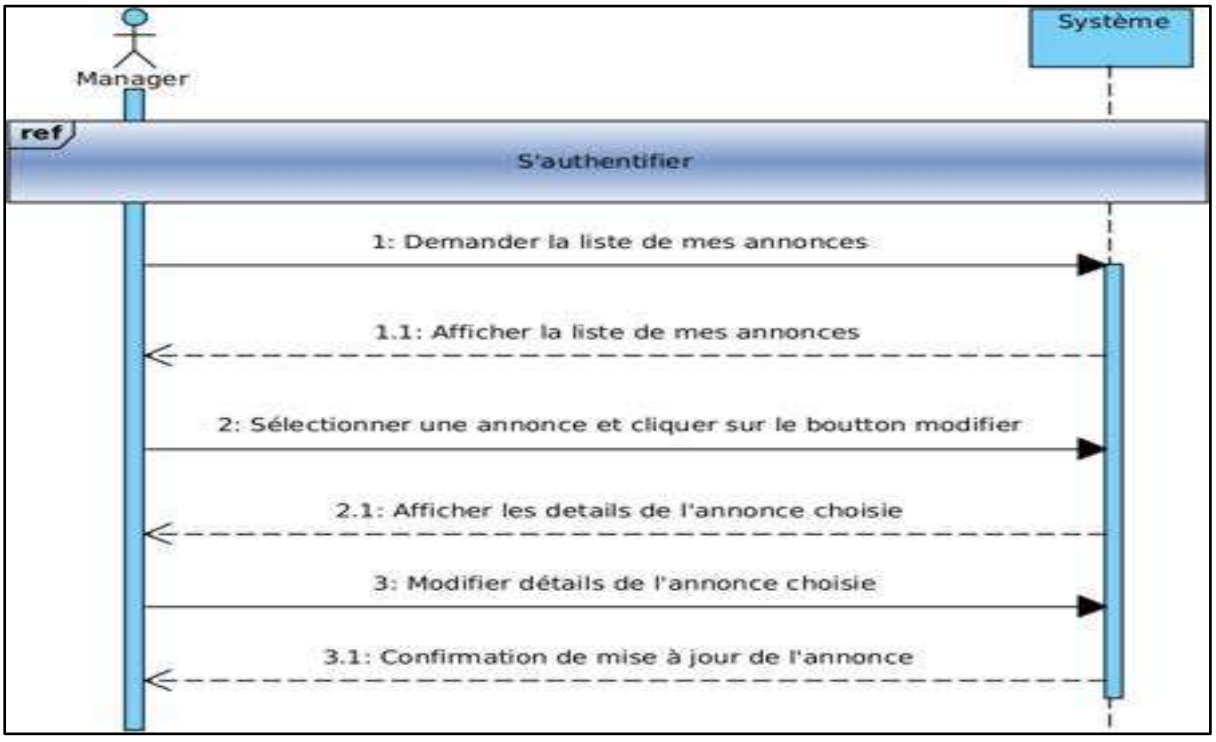

Figure 19 Diagramme de séquence « Modifier annonces »

Cas d'utilisation	Modifier annonces
Acteur	Manager
Intérêts	Il permet au manager de modifier une annonce.
Pré condition	Manager authentifié
Scénario nominal	1. Le manager demande la liste de ses annonces publiées. 2. Le système affiche les annonces publiéesdu manager. 3. Le manager choisit une annonce et cliquesur le bouton modifier. 4. Le système affiche les détails de l'annonce. 5. Le manager modifie l'annonce concernée. 6. Le système modifie l'annonce
Post condition	La liste des annonces d'abonné est mise à jour
Exception	Le système affiche un message d'erreur indiquant que le formulaire de modification d'annonce contient des champs vides.

Tableau 7 Description textuelle du diagramme de séquence « Modifier annonce »

2.4 Diagramme de séquence « Supprimer annonce »

Le tableau n° 8 et la figure n° 23 décrivent le diagramme de séquence «Supprimer annonce »

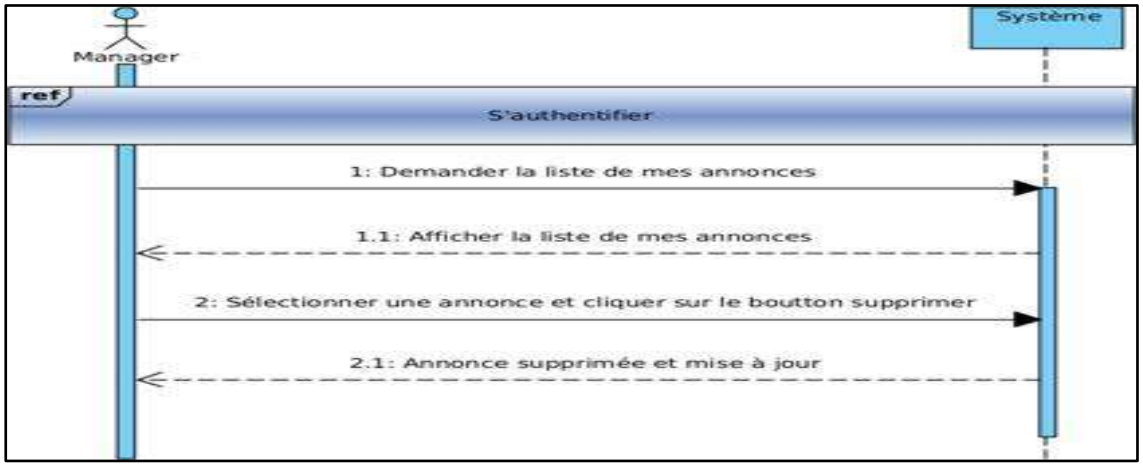

Figure 20 Diagramme de séquence « Supprimer annonce »

Cas d'utilisation	Supprimer annonce
Acteur	Manager
Intérêts	Il permet au manager de supprimer une annonce.
Pré condition	Manager authentifié
Scénario nominal	1. Le manager demande la liste de ses annonces publiées. 2. Le système affiche les annonces publiéesdu manager. 3. Le manager choisit une annonce et clique sur le Bouton supprimer. 4. Le système supprime l'annonce.
Post condition	La liste des annonces d'abonné est mise à jour
Exception	Le système affiche un message d'erreur indiquant que le formulaire de modification d'annonce contient des champs vides.

Tableau 8 Description textuelle du diagramme de séquence <<Supprimer annonce>>

Conclusion

Dans ce chapitre, nous avons présenté une vue conceptuelle du système à réaliser. Ainsi, nous avons défini le diagramme de cas d'utilisation générale, les diagrammes détaillés descas d'utilisation de notre projet, suivis de quelques diagrammes de séquences. Nous pouvons entamer la phase suivante qui est la phase de réalisation de la solution.

Chapitre 3. Conception de l'application

Introduction

I- Conception préliminaire des interfaces-Prototypes

II- Conception architecturale

III- Conception de base de données

IV- Conception détaillées

Conclusion

Introduction

Afin d'atteindre les objectifs de notre projet, et après l'analyse et la spécification desbesoins de notre future application, nous allons définir les différentes étapes de notreconception pour donner suite à notre méthodologie de travail.

I. Conception préliminaire des interfaces – Prototypes

Une maquette est un produit jetable donnant aux utilisateurs une vue concrète mais non définitive de la future interface de l'application. Elle est développée rapidement afin d'améliorer la relation développeur-client. Les maquettes de notre application sont les suivantes.

Les figures n° 24,25,26,27 28 et 29 illustrent des interfaces préliminaires de notre application.

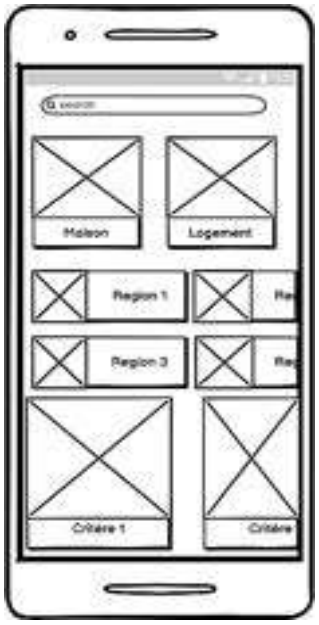

Figure 22 Maquette préliminaire de l'interface d'Accueil

Figure 21 Maquette préliminaire de l'interface «Rechercher par localisation»

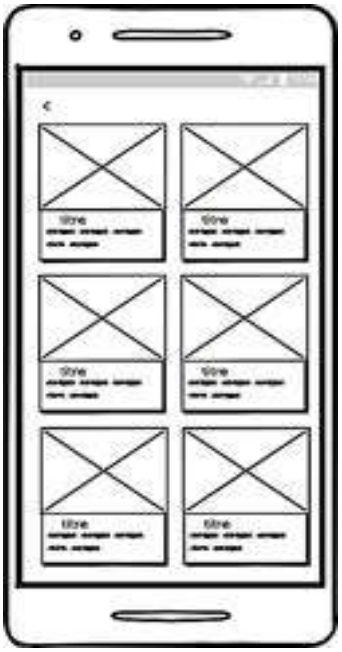

Figure 23 Maquette préliminaire de l'interface « Rechercher par critère

Figure 24 Maquette préliminaire de l'interface « House »

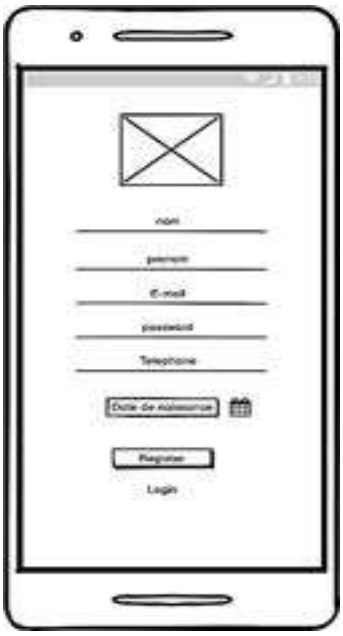

Figure 25 Maquette préliminaire de l'interface « Inscription »

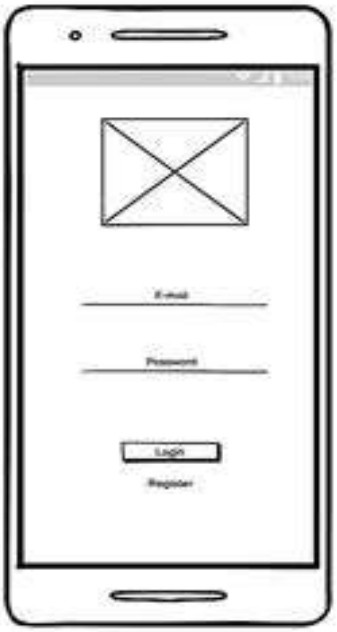

Figure 26 Maquette préliminaire de l'interface

II. Conception architecturale

Avant de développer notre application, il est indispensable de choisir un modèle de conception (pattern design). Parmi les patrons les plus connues, nous mentionnons lespatrons Modèle-Vue-Contrôleur (MVC), Modèle-Vue-Vue Modèle (MVVM).Mais, quelle est la différence entre eux ?

1.MVC

MVC est un modèle de conception de logiciel. Il pousse une bifurcation de préoccupations, ce qui signifie que le modèle et la logique du contrôleur sont séparés del'interface utilisateur (vue). Par conséquent, la maintenance et les tests de l'application deviennent simples et faciles. Le modèle de conception MVC divise une application en trois aspects principaux : Modèle, Vue et Contrôleur, comme indiqué dans la Figure 30 :

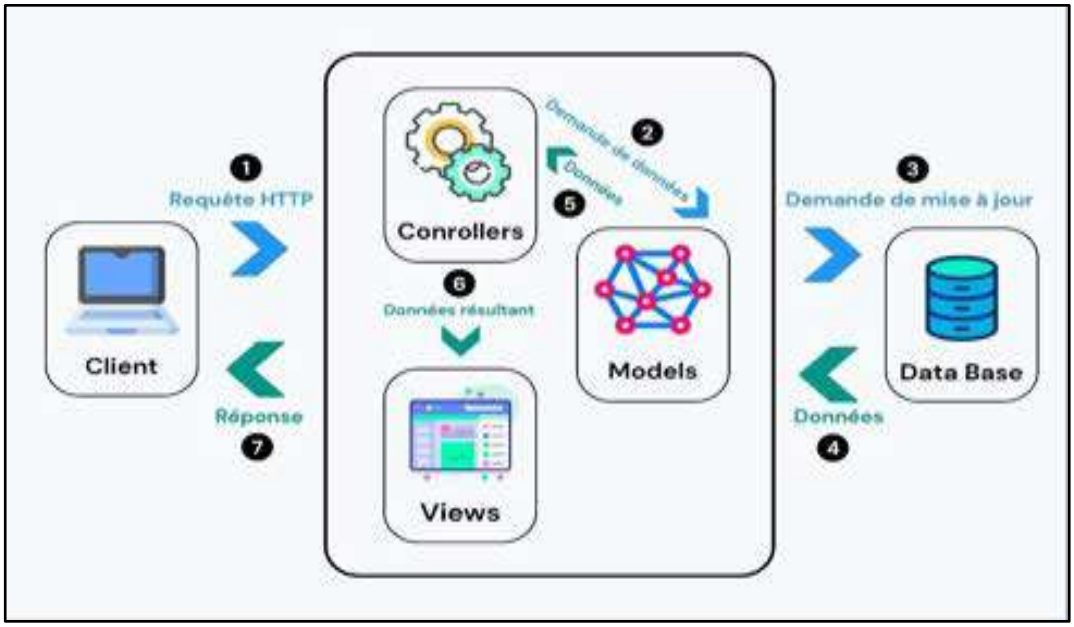

Figure 27 Architecture MVC

- **Model :** Le modèle représente une collection de classes qui explique la logique métier, à savoir le modèle métier et le modèle de données (opérations d'accès aux données). IL définit également les règles métier pour les moyens de données comme la façon dont les données peuvent être modifiées et manipulées.
- **View :** La vue représente les composants de l'interface utilisateur tels que CSS, jQuery, HTML, etc (dans le web). Elle affiche les données reçues du contrôleur comme résultat.

- **Controller :** La responsabilité du contrôleur est de traiter les demandes entrantes. Il obtient l'entrée des utilisateurs via la vue, puis traite les données de l'utilisateur via le modèle, en renvoyant les résultats à View. Il agit comme un médiateur entre la vue et le modèle.

En séparant le code d'une application mobile de cette manière, l'accès aux données est totalement séparé du reste de l'application. Sans dépendance à l'interface utilisateur ou à la logique des actions de l'utilisateur, il est bien plus simple d'écrire des tests automatisés.De plus, la duplication de code sera considérablement amoindrie.
Mais qu'en est-il de l'adaptation et de l'évolution de notre application ?
Si le modèle dedonnées est amené à changer, nous devons obligatoirement modifier l'ensemble de nos entités.

2.MVVM

Le modèle MVVM prend en charge la liaison de données bidirectionnelle entre View et ViewModel. Cela permet la propagation automatique des modifications, à l'intérieur de l'état de ViewModel à la vue. Généralement, ViewModel utilise le modèle d'observateur pour informer les modifications apportées au modèle View-Model au modèle. La Figure 31 illustre la structure générique d'une architecture MVVM.

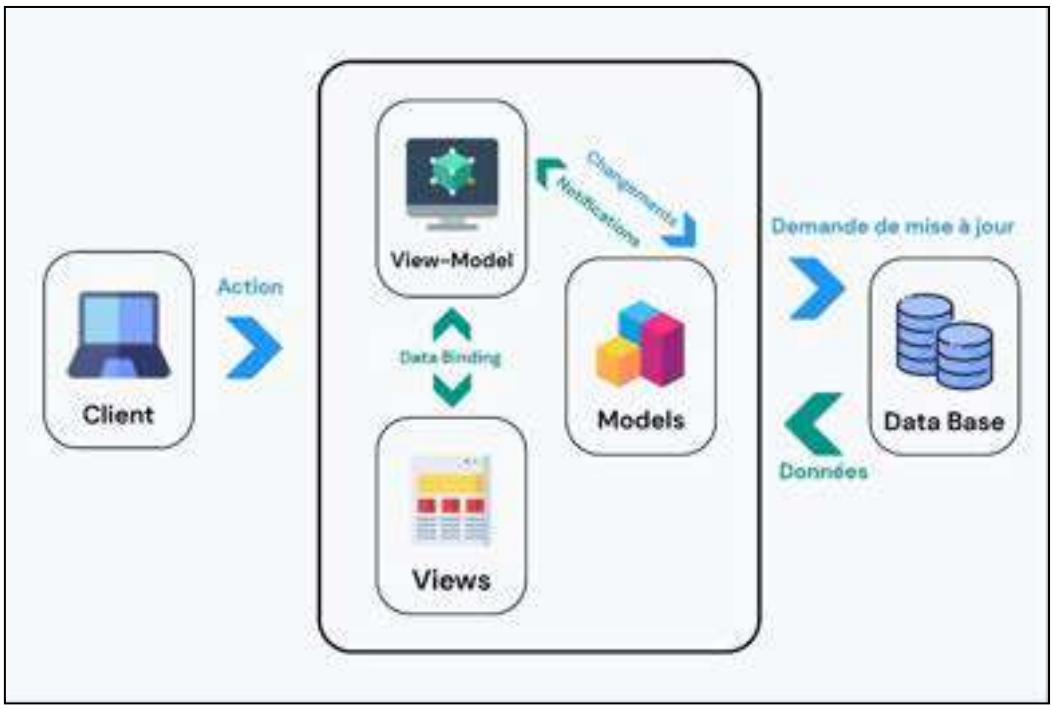

Figure 28 Le modèle MVVM

- **Model :** Le modèle représente une collection de classes qui explique le modèle métieret le modèle de données. Il définit également les règles métier.
- **Vue :** La vue représente les composants de l'interface utilisateur tels que CSS, jQuery,HTML, etc...
- **ViewModel :** Le modèle de vue est responsable de l'affichage des méthodes, Commandes et autres fonctions qui aident à maintenir l'état de la vue, à manipuler le modèle à la suite d'actions sur la vue et à déclencher les événements dans la vue elle-même.

Notre choix

D'après cette étude, nous avons choisi de travailler avec l'architecture MVC, car elle permet de bien séparer la partie logique de la partie présentation et convient parfaitementà la nature et dimension de notre projet. En effet, vue cette séparation entre les parties, il est très facile d'ajouter et de modifier le code d'une section sans que le reste ne soit toucher. C'est un pattern qui se prête très bien au développement des petites et moyennesapplications.

III. Conception de la base de données

1.Modèle relationnel des données

La figure 32 illustre le modèle relationnel des données

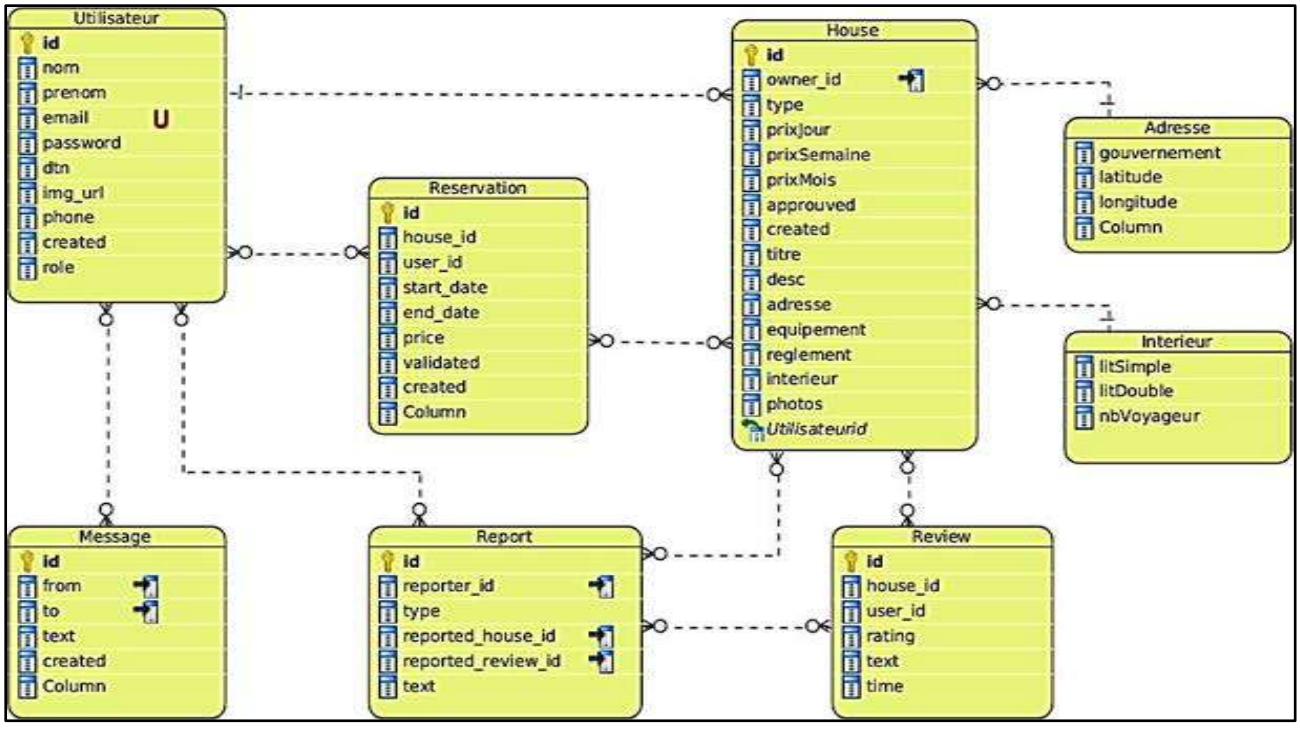

Figure 29 le modèle relationnel des données

2.Modèle de description de données

La figure 33 illustre le modèle de l'entité Utilisateur.

Figure 30 le modèle de l'entité Utilisateur

La figure 34 illustre le modèle de l'entité House.

Figure 31 le modèle de l'entité House

La figure 35 illustre le modèle de l'entité Message.

Figure 32 le modèle de l'entité Message

La figure 36 illustre le modèle de l'entité Réservation.

Figure 33 le modèle de l'entité Réservation

La figure 37 illustre le modèle de l'entité Review.

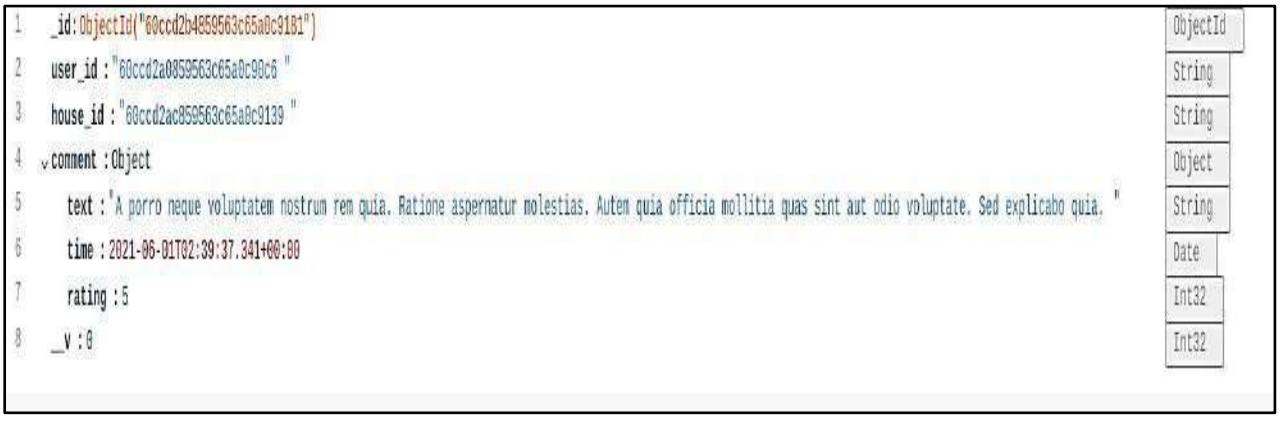

Figure 34 Modèle Review

IV. Conception détaillée

1. Vue statique

1.1 Diagramme de classe

Un diagramme de classe dans le langage UML est un type de diagramme de structurestatique qui décrit la structure d'un système en montrant le système de classes, leurs attributs, les opérations (ou) les méthodes et les relations entre les classes.

La Figure 38 illustre bien les diagrammes de classe de notre système selon le modèle MVC:

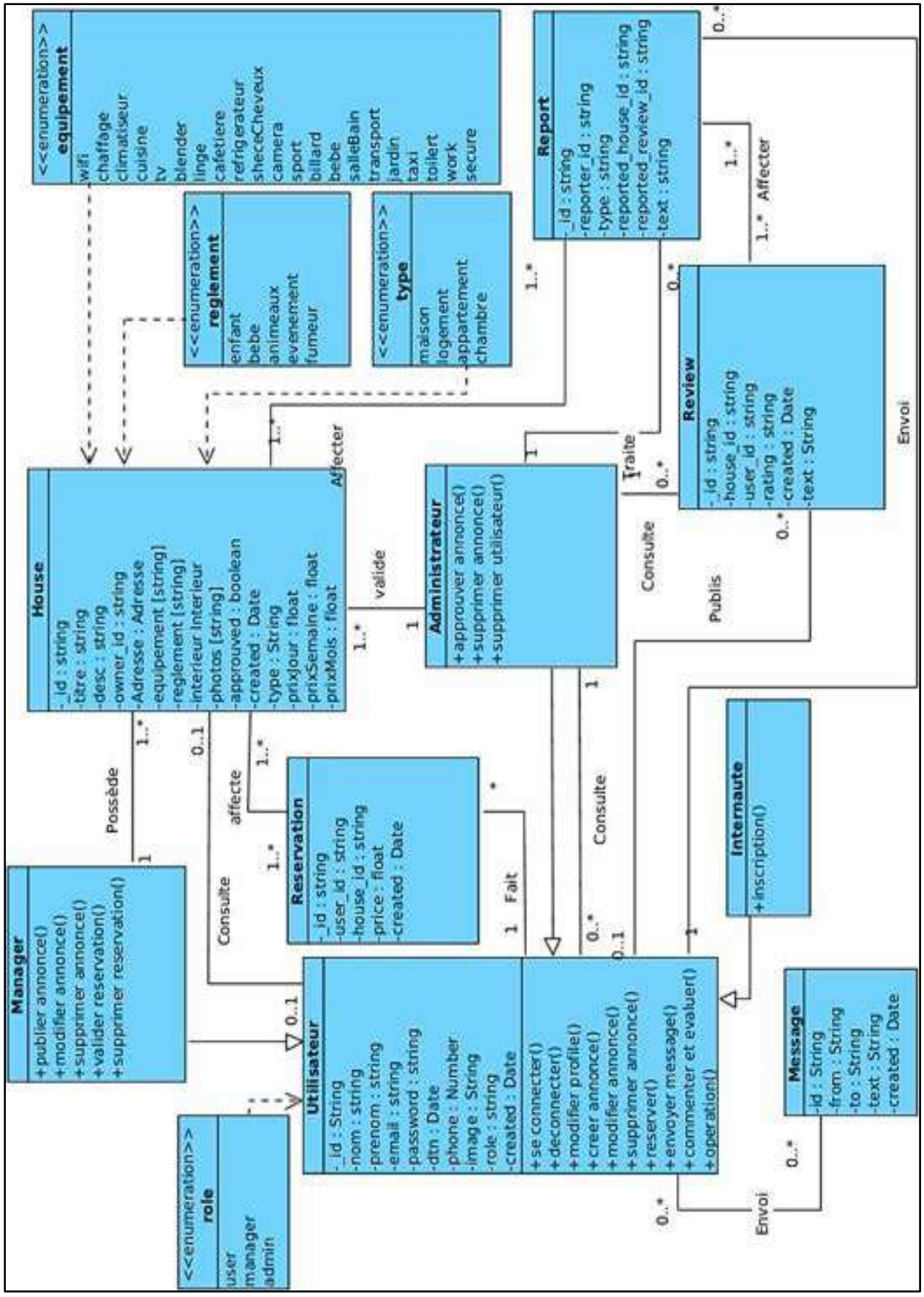

Figure 35 Diagramme de classe

2. Vue dynamique

Le diagramme de séquence dynamique, appelé aussi diagramme d'interaction en conception, permet de représenter les interactions d'objets dans le cadre d'un diagrammedes cas d'utilisation selon un ordre chronologique.

2.1 Diagramme d'interaction

La figure 39 représente le diagramme de séquence détaillé « Authentification ».

Ce diagramme montre les différentes étapes permettant à l'utilisateur d'approuvé son identité pour profiter de quelques services (réserver, commenter, évaluer, favoriser, communiquer, etc ...).

L'interaction commence par la demande de la page « Authentification ». Cette demande est envoyée vers le serveur qui la traite. La demande est envoyée vers la base de donnéesafin de vérifier l'existence de l'utilisateur en premier lieux puis en cas de succès une vérification de mot de passe se fait en comparant celle introduite par l'utilisateur avec celle qui est stocké et haché à la base de données.

L'utilisation ultérieure de l'application nécessite un module d'identification qui négligetoute authentification non nécessaire d'où ont généré un token qui se place à l'entête de chaque ultérieure requête.

Après, les données de l'utilisateur connecté doivent être enregistré à la mémoire internedu périphérique d'où vient le rôle de module Redux qui joue le rôle d'une variable globale qui stocke les informations qui seront accessible depuis quelle View de l'application.

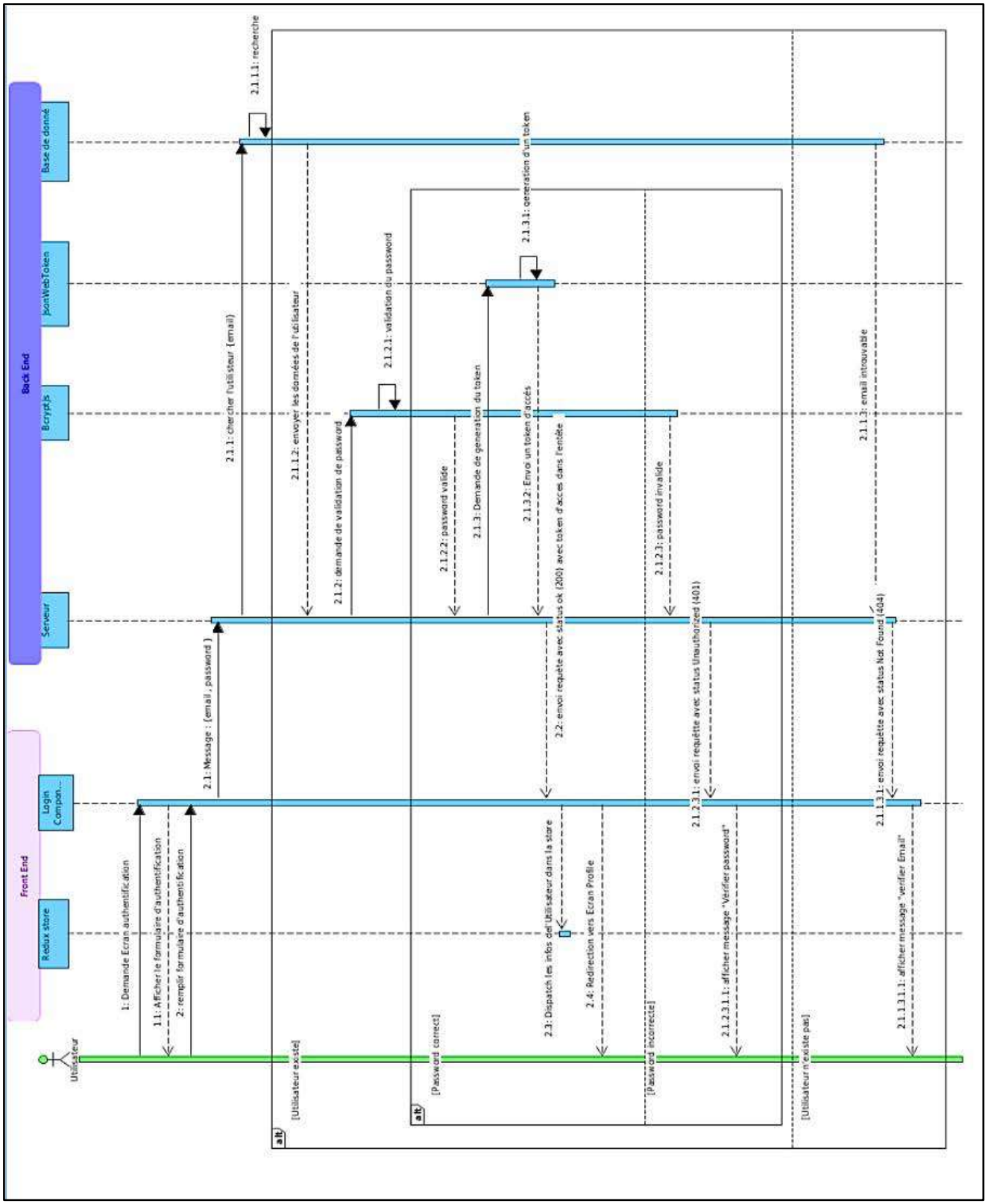

Figure 36 Diagramme d'interaction «S'authentifier»

Conclusion

Nous avons abordé dans ce chapitre la conception sur laquelle est basée le fonctionnementde notre application. D’abord, nous avons présenté l’architecture générale de notre application.

Ensuite, nous avons présenté la conception de notre application en utilisant, le diagramme d’entité association, le diagramme de classe, et les diagrammes de séquences. Le chapitrequi suit décrit la dernière phase du projet: la phase de réalisation qui englobe le choix des technologies utilisées en vue d’implémenter l’application.

Chapitre 4. Réalisation

Plan :

Introduction

I- Besoins techniques

II- Les interfaces de notre application

Conclusion

Introduction

Nous arrivons maintenant à la phase ultime. Cette dernière partie est la plus importante puisqu'elle met en réalité toute la théorie précédente.
Dans un premier temps nous présentons les outils techniques utilisées .Dans un second temps, l'environnement de réalisation sur le plan logiciel.et dans un troisième temps, nous illustrons la réalisation de notre travail en se basant sur quelques captures d'écran.

I. Besoins techniques

Nous nous intéressons ici à la branche droite du cycle en Y « Capture des besoins techniques» qui capitalise le savoir-faire technique. Afin de bien expliquer nos choix technologiques, nous avons fait recours à une étude comparative entre les différentes technologies qui peuvent être utilisées durant notre projet.

1. Choix technologique

Nous avons développé notre application avec la pile MERN (React Native/ReactJs, NodeJs, Express JS et MongoDB).
La figure 40 illustre la plupart des technologies utilisées durant le développement de notre application.

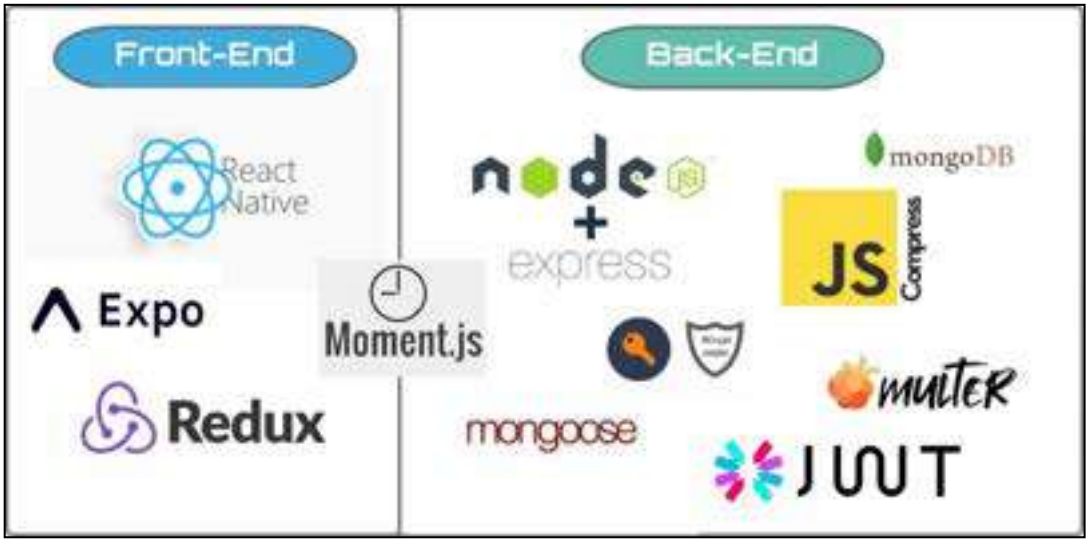

Figure 37 Choix technologique de notre application

Le Tableau 9 illustre l'utilisation et la description des technologies utilisées dans le

Dans cette section, nous présentons les différents environnements matériels et logiciels

Nom	Utilisation	Description
React-native	**Front**	React Native est un framework d'applications mobiles open source créé par Facebook. Il est utilisé pour développer des applications pour Android, iOS et UWP en permettant aux développeurs d'utiliser React avec les fonctionnalités native de ces plateformes
Expo	**Front**	Expo est un framework et une plate-forme pour les applications React universelles. Il s'agit d'un ensemble d'outils et de services construits autour deReact Native et des plates-formes natives qui vous aident à développer, créer, déployer et itérerrapidement sur iOS, Android et des applications web à partir de la méme base de code javScript/TypeScript
Redux	**Front**	Redux est une bibliothèque open-source JavaScript de gestion d'état pour applications web. Elle est plus couramment utilisée avec des bibliothèques comme React ou Angular pour la construction d'interfaces utilisateur. Semblable à l'architecture Flux, elle a été créée par Dan Abramov et Andrew Clark.
ReactJs	**Front**	React est une bibliothèque JavaScript libre développée par Facebook depuis 2013. Le but principal de cette bibliothèque est de faciliter la création d'application web monopage, via la création de composants dépendant d'un état et générant une page HTML à chaque changement d'état.
MomentJs	**Front+Back**	MomentJS est une bibliothèque JavaScript qui permet d'analyser, de valider, de manipuler et d'afficher la date/l'heure en JavaScript de manière très simple.

NodeJs	**Back**	Node.js est une plateforme logicielle libre en JavaScript, orientée vers les applications réseau événementielles hautement concurrentes qui doivent pouvoir monter en charge. Elle utilise la machine virtuelle V8, la librairie libuv pour sa boucle d'évènements, et implémente sous licence MIT les spécifications CommonJS.
ExpressJs	**Back**	Express.js est un framework pour construire des applications web basées sur Node.js. C'est de fait le framework standard pour le développement de serveur en Node.js.
mongoose	**Back**	Mongoose est une bibliothèque de modélisation de données d'objets (ODM) pour MongoDB et Node. js. Il gère les relations entre les données, fournit une validation de schéma et est utilisé pour traduire entre des objets dans le code et la représentation de ces objets dans MongoDB.
Multer	**Back**	Multer est un middleware node.js pour la gestion de multipart/form-data , qui est principalement utilisé pour télécharger des fichiers. Il est écrit sur le busboy pour une efficacité maximale.
Jwt	**Back**	JSON Web Token est un standard ouvert défini dansla RFC 7519. Il permet l'échange sécurisé de jetons entre plusieurs parties. Cette sécurité de l'échangese traduit par la vérification de l'intégrité des données à l'aide d'une signature numérique. Elle s'effectue par l'algorithme HMAC ou RSA.
Bcrypt	**Back**	Bcrypt est une fonction de hachage créée par NielsProvos et David Mazières. Elle est basée sur l'algorithme de chiffrement Blowfish et a été présentée lors de USENIX en 1999.

Tableau 9 Choix technologique de notre application

2.Environnement matériel

Pendant les différentes phases de notre projet, nous avons disposé d'un PC avec les caractéristiques figurant dans le tableau 10

Fournisseur	*Processeur*	*RAM*	*Système d'exploitation*
Lenovo	11th Gen Intel® Core™ i7-1165G7 @ 2.80GHz × 8	8 Go	Windows 10

Tableau 10 Environnement matériel

3.Les environnements de développement

LOGO	NOM	Description
	Visual Studio code	Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS.
	git	Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2.
	MongoDB Atlas	MongoDB Atlas est une base de données distribuée, universelle et basée sur des documents, qui a été conçue pour les développeurs d'applications modernes et pour l'ère du Cloud.

Tableau 11 Les environnements de dévloppenment

4.Environnement de conception

Visual Paradigm est un complément de bureau qui permet aux utilisateurs de d'incorporer des diagrammes interactifs et modifiables

Figure 38 logo Visual Paradigm

II. Les interfaces de notre application

1.Interfaces application mobile

Les figures présentent l'interface d'accueil de notre application.

Figure 40 Interface d'accueil 1

Figure 39 Interface d'accueil 2

La figure n°44 présente l'interface de recherche par localisation de notre application.

Figure 41 Interface de recherche par localisation

La figure n°45 présente l'interface du résultat de la recherche par localisation de notre application.

Figure 42 Interface du résultat de recherche par localisation

La figure n°46 présente l'interface du résultat de la recherche par type de notre application

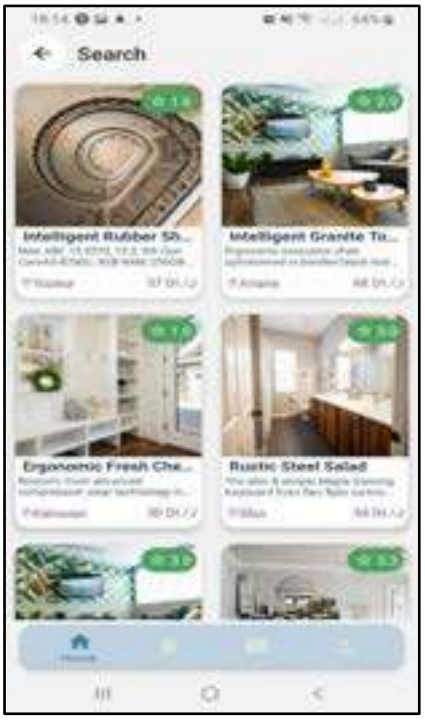

Figure 43 Interface du résultat de recherche

Les figures n° 47 et n°48 présentent l'interface d'une annonce.

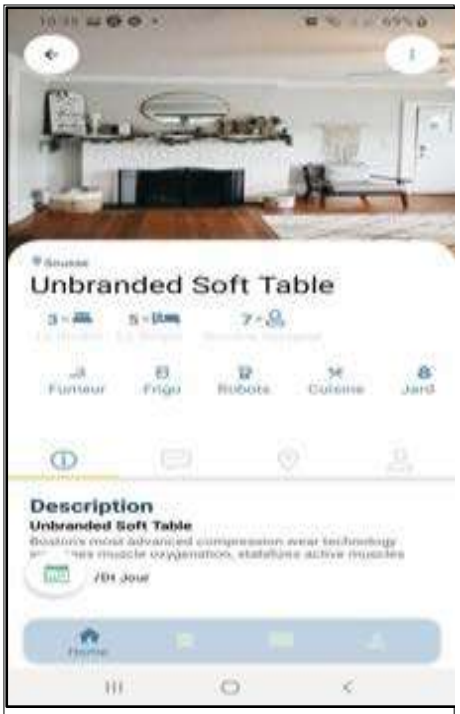

Figure 45 Interface principale d'une annonce

Figure 44 L'interface des commentaires d'une annonce

Les figures n° 49 et n°50 présentent les interfaces d'authentification et de s'inscrire.

Figure 47 Interface d'authentification

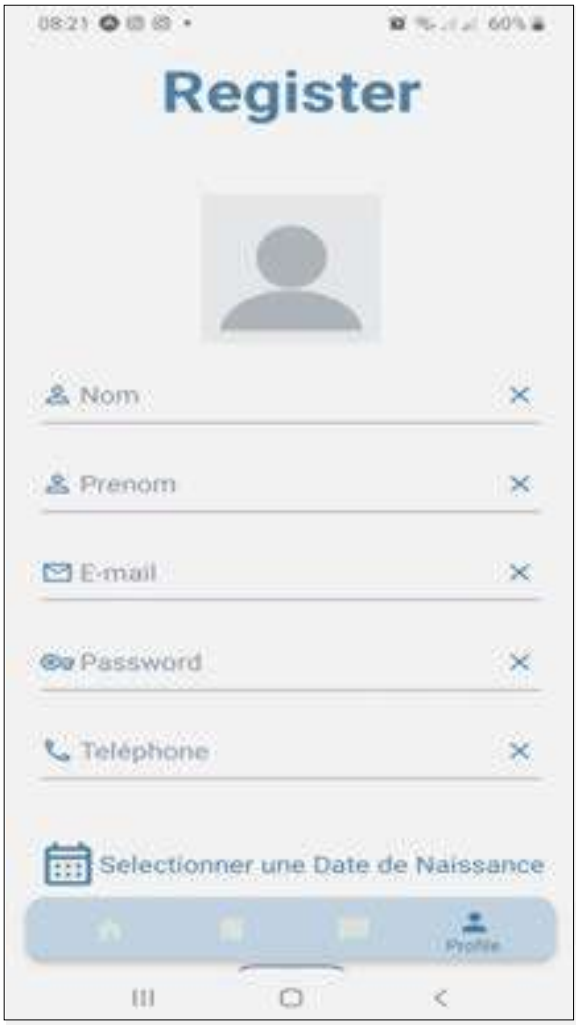

Figure 46 Interface de s'inscrire

La figure n°51 présente l'interface du profile de l'utilisateur.

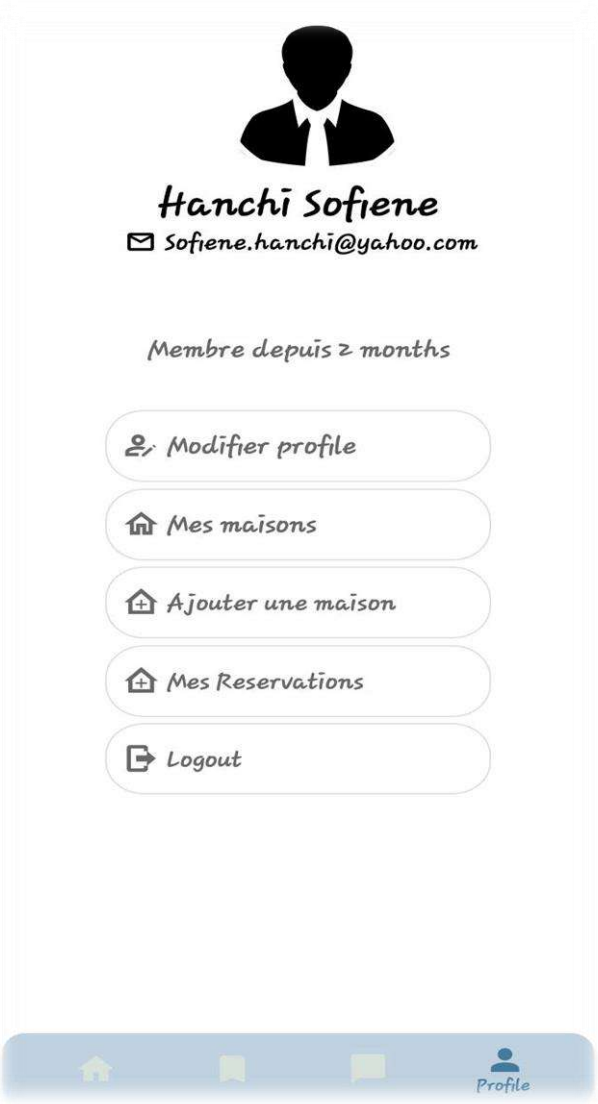

Figure 48 Interface du profile de l'utilisateur

Les figures n° 52 et n°53 présentent les interfaces des listes d'annonces et de réservations du manager.

Figure 49 Interface liste des annonces du manager

Figure 50 Interface liste des réservations des annonces du manager

Conclusion

Au cours de ce chapitre, nous avons invoqué notre environnement du travail et justifié nos choix technologiques. Nous avons ensuite donné un aperçu sur le travail réalisé à travers la visualisation des interfaces les plus significatives.

Conclusion Générale

Ce projet consiste à concevoir et développer une application mobile pour la gestion des réservations des logements en Tunisie qui facilite les démarches de réservations des séjours.

Pour pouvoir compléter notre mission, nous avons détaillé les différentes étapes d'analyse, de conception et de réalisation de cette application.

Le présent rapport s'est articulé dans quatre chapitres. Nous avons commencé par le chapitre « Contexte général et étude préalable » qui localise le contexte général de notre projet et qui est dédié à l'étude de l'existant où nous avons présenté une synthèse des solutions existantes sur le marché. Puis, nous avons formulé les principales fonctionnalités de la solution proposée.

Le deuxième chapitre « Analyse et spécification des besoins » qui permet de présenter la spécification des besoins fonctionnels, non fonctionnels et techniques, ainsi les diagrammes de cas d'utilisation, les diagrammes d'activité et les diagrammes de séquences système. A la lumière de ce chapitre, nous avons entamé le troisième chapitre intitulé « Conception », dans lequel nous avons présenté la notation de modélisation utilisée ainsi que l'ensemble des diagrammes conçus.

Le quatrième chapitre « Réalisation » qui a été consacré à l'étude technique où nous avons détaillé notre environnement de travail suivi d'une présentation des différentes fonctionnalités de notre « Plateforme de gestion des réservations des logements » à travers des captures d'écran.

Perspective

Comme tout autre travail, notre projet ne peut pas prétendre la perfection, il y a quelques améliorations que nous puissions les ajouter comme une Dashboard web qui permet la gestion de l'application mobile :

- Gère les annonces et leurs utilisateurs
- Consulter et bloque les utilisateurs
- Gère les réservations

Bibliographie

[1] https://www.ambient-it.net/react-native-vs-xamarin-vs-ionic-vs-flutter/ , consulté le 05/05//2022

[2] https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/ , consulté le 10/06/2022

[3] https://www.silicon.fr/developpeurs-10-langages-programmation-apprecies-333211.html, consulté le 10/06/2022

Résume

Ce projet consiste à concevoir et développer une application mobile pour la gestion des réservations des logements en Tunisie qui facilite les démarches de réservation des séjours. Beaucoup d'opportunités seront offertes via l'application.

Mots clés: React Native, React Js, Redux, Node JS, Express JS, MongoDB, mongoose, AWS, Auth, UML.

ABSTRACT

This project consists in designing and developing a mobile application for the management of accommodation reservations in Tunisia which facilitates the process of booking stays.Lots of opportunities will be offered through the app.

Keywords: React Native, React Js, Redux, Node JS, Express JS, MongoDB, mongoose, AWS , Auth0, UML.