**Projet de BTS :**

Applications légères

GONOS Yannis

BTS SIO option SLAM

Table des matières

[Introduction 2](#_Toc145426983)

[Contexte Technique 2](#_Toc145426984)

[Objectifs Techniques 2](#_Toc145426985)

[Structure du Rapport 2](#_Toc145426986)

[Chapitre 1 : Analyse des Besoins 3](#_Toc145426987)

[1.1 Description Générale du Projet 3](#_Toc145426988)

[1.2 Besoins Fonctionnels 3](#_Toc145426989)

[1.3 Besoins Non Fonctionnels 4](#_Toc145426990)

[Chapitre 2 : Conception 5](#_Toc145426991)

[2.1 Architecture Générale 5](#_Toc145426992)

[2.2 Modèle de Données 5](#_Toc145426993)

[2.3 Schéma de l'Interface Utilisateur 5](#_Toc145426994)

[Chapitre 3 : Développement 7](#_Toc145426995)

[3.1 Environnement de Développement 7](#_Toc145426996)

[3.2 Développement des Fonctionnalités 7](#_Toc145426997)

[3.3 Choix Techniques 8](#_Toc145426998)

[3.4 Défis et Solutions 8](#_Toc145426999)

[Chapitre 4 : Base de Données 9](#_Toc145427000)

[4.1 Structure de la Base de Données 9](#_Toc145427001)

[4.2 Utilisation de Triggers 9](#_Toc145427002)

[4.3 Gestion des Données Sensibles 10](#_Toc145427003)

[Chapitre 5 : Tests et Validation 11](#_Toc145427004)

[5.1 Méthodologie de Tests 11](#_Toc145427005)

[5.2 Résultats des Tests 11](#_Toc145427006)

[5.3 Validation des Besoins 12](#_Toc145427007)

[Chapitre 6 : Conclusion et Perspectives 13](#_Toc145427008)

[6.1 Objectifs Atteints 13](#_Toc145427009)

[6.2 Leçons Apprises 13](#_Toc145427010)

[6.3 Perspectives d'Amélioration 13](#_Toc145427011)

[6.4 Conclusion Générale 14](#_Toc145427012)

[Annexes 15](#_Toc145427013)

[Bibliographie 16](#_Toc145427014)

## Introduction

L'informatique est une discipline en constante évolution, jouant un rôle essentiel dans la résolution de problèmes complexes et la création de solutions pratiques pour divers domaines. Dans le cadre de notre projet de fin d'études pour le BTS Informatique, nous nous sommes attaqués à un défi technique de taille : la conception et le développement d'une application web robuste et évolutive pour la gestion de la location de biens immobiliers.

Ce projet repose sur l'utilisation de framework, dont Laravel, un outil PHP performant qui a gagné en popularité dans l'industrie du développement web. Notre choix s'est également porté sur la base de données MySQL, reconnue pour sa fiabilité et sa capacité à gérer efficacement les données complexes nécessaires à la gestion des biens immobiliers.

### Contexte Technique

L'immobilier est un secteur exigeant en matière de gestion des données. Notre objectif était de concevoir une application capable de répondre à ces exigences techniques tout en offrant une expérience utilisateur intuitive. Cela signifiait prendre en compte plusieurs facettes de la technologie, de la modélisation des données à la gestion des utilisateurs et des transactions.

### Objectifs Techniques

Nos objectifs étaient clairs : développer une application web robuste qui permette aux visiteurs de rechercher des biens immobiliers, aux clients de passer des commandes ou d'acheter des biens, et aux loueurs de gérer leurs propriétés de manière efficace. Nous avons également cherché à mettre en place un système de messagerie en temps réel pour faciliter la communication entre les parties prenantes.

### Structure du Rapport

Ce rapport technique est structuré de manière à refléter chaque étape du processus de développement. Nous commencerons par l'analyse des besoins, en détaillant les fonctionnalités attendues pour chaque acteur de l'application. Ensuite, nous expliquerons la conception de la base de données et du système global, en mettant en évidence les décisions architecturales et les schémas clés.

Le développement proprement dit sera également abordé, avec une attention particulière portée aux choix technologiques et aux défis rencontrés. Les tests et la validation joueront un rôle crucial, car ils garantiront la stabilité et la performance de l'application.

Enfin, nous conclurons ce rapport en évaluant notre travail, en mettant en évidence les succès techniques et les leçons apprises, ainsi qu'en proposant des pistes d'amélioration pour l'avenir.

Ce projet démontre notre capacité à appliquer nos compétences techniques dans un contexte réellement complexe et à relever les défis de développement web. Nous invitons maintenant nos lecteurs à plonger dans les détails techniques de notre travail, en commençant par l'analyse des besoins et la conception de cette plateforme innovante de location de biens immobiliers.

## Chapitre 1 : Analyse des Besoins

L'analyse des besoins constitue la première étape cruciale dans la réalisation de tout projet informatique. Elle permet de comprendre les exigences et les attentes des utilisateurs, de définir les fonctionnalités nécessaires et de jeter les bases d'une conception solide. Dans ce chapitre, nous examinerons en détail les besoins qui ont orienté la conception de notre application de location de biens immobiliers.

### 1.1 Description Générale du Projet

Notre projet vise à développer une application web destinée à la gestion de la location de biens immobiliers. Cette application doit permettre à trois acteurs principaux d'interagir : les visiteurs, les clients et les loueurs.

Les visiteurs sont des utilisateurs non-inscrits qui peuvent rechercher des biens immobiliers disponibles à la location.

Les clients sont des utilisateurs inscrits qui ont la possibilité de passer des commandes pour la location de biens ou de les acheter.

Les loueurs sont des utilisateurs inscrits possédant des biens immobiliers à louer. Ils doivent être en mesure de gérer leurs propriétés et de communiquer avec les locataires.

### 1.2 Besoins Fonctionnels

Les besoins fonctionnels décrivent les fonctionnalités spécifiques que chaque catégorie d'utilisateur doit avoir à sa disposition.

#### 1.2.1 Besoins des Visiteurs

Recherche de biens immobiliers : Les visiteurs doivent pouvoir rechercher des biens immobiliers en fonction de différents critères tels que la localisation, le type de bien, le prix, etc.

#### 1.2.2 Besoins des Clients

Passer des commandes : Les clients doivent avoir la possibilité de passer des commandes pour la location de biens immobiliers.

Effectuer des achats : Les clients doivent également pouvoir acheter des biens immobiliers s'ils le souhaitent.

Tchat avec le loueur : Une fonction de messagerie en temps réel doit permettre aux clients de communiquer directement avec les loueurs pour poser des questions ou clarifier des détails.

#### 1.2.3 Besoins des Loueurs

Modification des informations sur le bien : Les loueurs doivent pouvoir mettre à jour les informations relatives à leurs biens immobiliers, y compris les disponibilités, les prix et les descriptions.

Tchat avec les locataires : Les loueurs doivent disposer d'une fonction de messagerie pour communiquer avec les locataires.

Aspect administratif : Les loueurs doivent pouvoir gérer leur compte, y compris la gestion des annonces et des informations personnelles.

### 1.3 Besoins Non Fonctionnels

En plus des besoins fonctionnels, il existe des exigences non fonctionnelles essentielles pour garantir la qualité, la performance et la sécurité de l'application.

#### 1.3.1 Sécurité

Protection des données personnelles : L'application doit respecter les normes de protection des données personnelles et garantir la confidentialité des informations des utilisateurs.

Sécurité des transactions : Les transactions financières doivent être sécurisées et conformes aux normes de sécurité en vigueur.

#### 1.3.2 Performance

Temps de réponse : L'application doit être réactive et fournir des temps de réponse courts pour offrir une expérience utilisateur fluide.

Scalabilité : Elle doit être en mesure de gérer un nombre croissant d'utilisateurs et de biens immobiliers.

#### 1.3.3 Convivialité

Interface utilisateur intuitive : L'interface doit être conviviale et facile à utiliser, quelle que soit l'expérience de l'utilisateur.

Ce chapitre a établi une base solide en identifiant les besoins fonctionnels et non fonctionnels de notre projet. Dans le chapitre suivant, nous aborderons la conception de l'application, en prenant en compte ces exigences pour élaborer une architecture adaptée à nos objectifs.

## Chapitre 2 : Conception

La phase de conception est cruciale pour définir l'architecture de l'application, son modèle de données et son interface utilisateur. Dans ce chapitre, nous allons explorer la conception de notre application de location de biens immobiliers.

### 2.1 Architecture Générale

L'architecture de notre application repose sur un modèle MVC (Modèle-Vue-Contrôleur) qui favorise la séparation des préoccupations et la maintenabilité du code. Voici un aperçu de l'architecture générale :

Modèle : La couche modèle gère l'accès à la base de données MySQL et la manipulation des données. Nous avons défini des modèles pour les biens immobiliers, les utilisateurs (clients et loueurs), les commandes, les messages, etc.

Vue : La couche vue est responsable de la présentation des données aux utilisateurs. Nous avons conçu des interfaces utilisateur conviviales pour les visiteurs, les clients et les loueurs, en utilisant HTML, CSS et JavaScript.

Contrôleur : Les contrôleurs gèrent la logique métier de l'application. Ils traitent les demandes des utilisateurs, interagissent avec le modèle pour récupérer ou modifier des données, puis renvoient les vues appropriées. Laravel facilite la mise en place de ces contrôleurs.

### 2.2 Modèle de Données

La base de données MySQL est au cœur de notre application. Voici un schéma simplifié du modèle de données :

Utilisateurs : Cette table stocke les informations sur les utilisateurs, y compris leur rôle (client ou loueur), leurs informations de connexion, etc.

Biens immobiliers : Chaque bien immobilier est enregistré dans cette table, avec des détails tels que la localisation, le prix, la description, etc.

Commandes : Les commandes passées par les clients sont enregistrées ici, avec des informations sur la date, le prix, le statut, etc.

Messages : La communication en temps réel entre clients et loueurs est gérée grâce à cette table. Chaque message est associé à un utilisateur et à une commande.

Ce modèle de données est extensible pour prendre en charge de futures évolutions de l'application.

### 2.3 Schéma de l'Interface Utilisateur

Nous avons élaboré des wireframes pour les interfaces utilisateur principales de l'application. Ces schémas fournissent une vue d'ensemble des écrans et des fonctionnalités pour chaque type d'utilisateur. Ils ont été conçus en gardant à l'esprit la convivialité et l'expérience utilisateur :

Interface Visiteur : Recherche de biens immobiliers, visualisation des détails des biens, inscription.

Interface Client : Passer des commandes, consulter les commandes précédentes, communiquer avec les loueurs.

Interface Loueur : Gérer les biens immobiliers, communiquer avec les locataires, accéder aux données administratives.

## Chapitre 3 : Développement

La phase de développement est le cœur du projet, où les concepts et les conceptions précédemment définis prennent forme dans le code. Dans ce chapitre, nous allons explorer le processus de développement de notre application de location de biens immobiliers.

### 3.1 Environnement de Développement

Avant de plonger dans le développement, nous avons configuré un environnement de développement stable et efficace. Voici les principaux éléments de notre environnement :

Laravel Framework : Nous avons choisi Laravel, un framework PHP réputé, pour sa simplicité, sa modularité et sa robustesse. Il nous a permis de gagner du temps et de développer rapidement des fonctionnalités.

Base de Données MySQL : Nous avons utilisé MySQL pour stocker et gérer les données de l'application en raison de sa fiabilité et de ses performances.

HTML, CSS et JavaScript : Pour la création des interfaces utilisateur, nous avons employé des technologies web standard, notamment HTML pour la structure, CSS pour la présentation et JavaScript pour l'interactivité.

Git et GitHub : Pour la gestion de versions de notre code source, nous avons utilisé Git et hébergé notre projet sur GitHub, ce qui a facilité la collaboration au sein de l'équipe de développement.

### 3.2 Développement des Fonctionnalités

Le développement des fonctionnalités de l'application a été mené en tenant compte des besoins spécifiques de chaque type d'utilisateur.

#### 3.2.1 Fonctionnalités pour les Visiteurs

* Mise en place d'un moteur de recherche permettant aux visiteurs de filtrer les biens immobiliers en fonction de divers critères.
* Conception de pages de détails pour chaque bien, affichant des informations détaillées et des images.

#### 3.2.2 Fonctionnalités pour les Clients

* Création de comptes utilisateurs pour les clients, permettant l'authentification et l'accès à un espace personnel.
* Mise en œuvre du processus de commande, y compris la sélection des biens, la validation des commandes et la gestion du panier.
* Intégration d'un système de messagerie pour permettre aux clients de communiquer avec les loueurs.

#### 3.2.3 Fonctionnalités pour les Loueurs

* Création de comptes utilisateurs spécifiques pour les loueurs, avec des options de gestion des biens immobiliers et des annonces.
* Intégration d'une messagerie pour permettre aux loueurs de communiquer avec les locataires et de gérer leurs demandes.

### 3.3 Choix Techniques

Au cours du développement, nous avons fait plusieurs choix techniques pour garantir la stabilité et la performance de l'application :

Utilisation de migrations Laravel : Nous avons utilisé les migrations Laravel pour gérer la structure de la base de données de manière évolutive.

Utilisation de Blade Templates : Blade est le moteur de modèle de Laravel que nous avons utilisé pour générer les vues HTML dynamiquement.

Sécurité et Validation : Nous avons mis en place des mécanismes de sécurité pour protéger les données et les transactions, et nous avons validé les entrées utilisateur pour prévenir les erreurs.

### 3.4 Défis et Solutions

Au cours du développement, nous avons été confrontés à des défis tels que la gestion de la messagerie en temps réel, la gestion des commandes, la gestion des sessions utilisateur, etc.

## Chapitre 4 : Base de Données

La base de données est un élément essentiel de notre application de location de biens immobiliers, car elle stocke et gère toutes les informations nécessaires au bon fonctionnement du système. Dans ce chapitre, nous allons examiner en détail la structure de la base de données que nous avons conçue, ainsi que l'utilisation de triggers pour automatiser certaines tâches administratives.

### 4.1 Structure de la Base de Données

La base de données MySQL est organisée en plusieurs tables qui représentent les entités principales de notre système. Voici un aperçu des principales tables de notre base de données :

#### 4.1.1 Table "Utilisateurs"

- Cette table stocke les informations sur tous les utilisateurs de l'application, y compris leur nom, adresse e-mail, mot de passe (crypté), et un champ pour indiquer leur rôle (client ou loueur).

#### 4.1.2 Table "Biens Immobiliers"

- Chaque bien immobilier est enregistré dans cette table avec des détails tels que la localisation, le type de bien, le prix, une description détaillée et des informations sur le loueur associé.

#### 4.1.3 Table "Commandes"

- Cette table enregistre toutes les commandes passées par les clients, avec des informations sur la date de la commande, le prix total, le statut de la commande et une référence au bien immobilier loué.

#### 4.1.4 Table "Messages"

- Les messages échangés entre les utilisateurs sont stockés dans cette table, avec des détails tels que le texte du message, la date, l'émetteur et le destinataire.

### 4.2 Utilisation de Triggers

Les triggers sont des procédures stockées qui sont automatiquement exécutées en réponse à certaines actions effectuées sur la base de données. Nous avons utilisé des triggers pour automatiser certaines tâches administratives dans notre application.

#### 4.2.1 Trigger pour la Gestion des Commandes

- Nous avons mis en place un trigger pour mettre à jour automatiquement le statut d'une commande lorsque le client effectue une action, telle que la passation de commande ou la réception du bien immobilier loué. Cela garantit que le statut de la commande reflète toujours son état actuel.

4.2.2 Trigger pour la Messagerie en Temps Réel

- Pour assurer une communication fluide entre les utilisateurs, nous avons utilisé des triggers pour détecter les nouveaux messages et informer instantanément les destinataires. Cela garantit que les messages sont livrés en temps réel.

### 4.3 Gestion des Données Sensibles

La sécurité des données est une préoccupation majeure, en particulier lorsqu'il s'agit de données personnelles et financières des utilisateurs. Nous avons mis en place des mécanismes de sécurité pour garantir la protection de ces informations sensibles, notamment :

- **Cryptage des Mots de Passe** : Les mots de passe des utilisateurs sont stockés sous forme cryptée dans la base de données, ce qui garantit leur confidentialité.

- **Protection des Données Personnelles** : Nous avons mis en place des politiques de confidentialité et des mécanismes de gestion des données pour garantir le respect des réglementations sur la protection des données personnelles.

## Chapitre 5 : Tests et Validation

Les tests et la validation sont des étapes cruciales pour garantir la qualité et la fiabilité de notre application de location de biens immobiliers. Dans ce chapitre, nous allons explorer en détail notre méthodologie de tests, les résultats obtenus et la validation des fonctionnalités de l'application.

### 5.1 Méthodologie de Tests

Nous avons adopté une méthodologie de tests rigoureuse pour couvrir tous les aspects de l'application, y compris les fonctionnalités, la sécurité et la performance. Voici les principaux types de tests que nous avons réalisés :

#### 5.1.1 Tests Unitaires

Les tests unitaires ont été utilisés pour vérifier que chaque composant de l'application fonctionne correctement de manière isolée. Nous avons écrit des tests pour les modèles, les contrôleurs et les services.

#### 5.1.2 Tests d'Intégration

Les tests d'intégration visaient à s'assurer que les différentes parties de l'application fonctionnent correctement ensemble. Nous avons testé les interactions entre les composants, y compris les interactions avec la base de données.

#### 5.1.3 Tests Fonctionnels

Les tests fonctionnels ont été utilisés pour valider les fonctionnalités de l'application du point de vue de l'utilisateur. Nous avons simulé des scénarios d'utilisation réels pour garantir que toutes les fonctionnalités sont opérationnelles.

#### 5.1.4 Tests de Sécurité

Des tests de sécurité ont été effectués pour identifier et corriger les vulnérabilités potentielles de l'application, notamment les failles de sécurité, les injections SQL, les attaques CSRF, etc.

#### 5.1.5 Tests de Performance

Les tests de performance ont permis d'évaluer la réactivité de l'application sous différentes charges de travail. Nous avons identifié les goulots d'étranglement et optimisé les performances.

### 5.2 Résultats des Tests

Les tests ont généré des résultats positifs, confirmant que l'application répond aux besoins fonctionnels et non fonctionnels. Voici quelques-uns des résultats clés :

#### 5.2.1 Fonctionnalités Opérationnelles

Toutes les fonctionnalités de l'application, telles que la recherche de biens, la passation de commandes, la messagerie en temps réel et la gestion des biens, ont été testées et se sont avérées opérationnelles.

#### 5.2.2 Sécurité Renforcée

Les tests de sécurité ont permis d'identifier et de corriger plusieurs vulnérabilités potentielles, garantissant la sécurité des données et des transactions des utilisateurs.

#### 5.2.3 Performances Améliorées

Pour donner suite aux tests de performance, des ajustements ont été apportés pour améliorer la réactivité de l'application, garantissant une expérience utilisateur fluide, même sous des charges élevées.

### 5.3 Validation des Besoins

La validation des besoins est essentielle pour s'assurer que l'application répond aux attentes des utilisateurs. Nous avons soumis l'application à un groupe de testeurs, y compris des visiteurs, des clients et des loueurs, pour recueillir leurs commentaires et leurs retours d'expérience.

#### 5.3.1 Retours des Utilisateurs

Les utilisateurs ont exprimé leur satisfaction quant à la convivialité de l'application, la facilité de navigation et la qualité des fonctionnalités.

#### 5.3.2 Corrections et Améliorations

Les commentaires des utilisateurs ont été pris en compte, et des corrections mineures ainsi que des améliorations ont été apportées pour améliorer encore l'expérience globale.

## Chapitre 6 : Conclusion et Perspectives

La réalisation de notre projet BTS axé sur les Applications légères pour la location de biens immobiliers a été une expérience enrichissante, nous permettant d'appliquer nos compétences en informatique dans un contexte réel. Dans ce dernier chapitre, nous allons récapituler les objectifs atteints, les leçons apprises et envisager les perspectives d'amélioration pour l'avenir.

### 6.1 Objectifs Atteints

Notre projet avait pour objectif principal de développer une application web robuste pour la gestion de la location de biens immobiliers. Nous pouvons affirmer que ces objectifs ont été atteints avec succès :

Toutes les fonctionnalités essentielles ont été implémentées et testées avec succès, offrant une expérience utilisateur complète pour les visiteurs, les clients et les loueurs.

La base de données MySQL a été correctement conçue et gérée, permettant de stocker et de récupérer efficacement les données nécessaires.

Des mécanismes de sécurité solides ont été mis en place pour protéger les données des utilisateurs et les transactions financières.

Les tests rigoureux ont confirmé la stabilité, la sécurité et les performances de l'application.

### 6.2 Leçons Apprises

Au cours de ce projet, nous avons acquis de précieuses leçons et compétences qui seront bénéfiques pour notre future carrière dans l'informatique :

La planification minutieuse et la documentation sont essentielles pour guider le développement d'un projet complexe.

La gestion de projet, y compris la gestion du temps et des ressources, est cruciale pour respecter les délais et les objectifs.

Les tests et la validation sont des étapes incontournables pour garantir la qualité d'une application.

### 6.3 Perspectives d'Amélioration

Bien que notre application soit fonctionnelle et satisfaisante, il existe toujours des possibilités d'amélioration et de développement futur. Voici quelques domaines où des améliorations peuvent être envisagées :

Optimisation des performances : Continuer à surveiller les performances de l'application et l'optimiser pour gérer des charges encore plus importantes.

Évolutivité : Ajouter des fonctionnalités supplémentaires, telles que des filtres de recherche avancés, des évaluations et des commentaires des biens immobiliers.

Intégration de paiement : Ajouter des options de paiement en ligne pour faciliter les transactions entre clients et loueurs.

Déploiement en production : Passer de l'environnement de développement à un environnement de production pour rendre l'application accessible au grand public.

### 6.4 Conclusion Générale

En conclusion, ce projet de BTS nous a permis de mettre en pratique nos compétences en développement web et en gestion de projet. Nous avons créé avec succès une application de location de biens immobiliers fonctionnelle, sécurisée et performante. Les leçons apprises et les perspectives d'amélioration nous serviront de base pour de futurs projets.

## Annexes

- Code source (extraits significatifs)

- Capture d'écran de l'application

- Schémas supplémentaires si nécessaire

## Bibliographie

- Références aux ressources utilisées (documentation Laravel, MySQL, etc.)