

**Institut Supérieur du Maroc du Groupe IISGA**

**Rapport de Projet de Fin d'Etudes**

Présenté en vue de l'obtention :  
**Du diplôme Technicien Spécialisé en Développement Informatique**

**SUJET :**

**Système de Gestion pour une Clinique Médicale**

Préparé sous la direction de :

Mr HISAIN

Présenté et soutenu publiquement par :

OMAR HAJJOUR

# Remerciement

C'est un grand plaisir que nous réservons cette page en signe de notre sincère gratitude envers tous ceux qui nous ont aidés de près ou de loin au bon déroulement de ce travail.

Nous tenons à remercier spécialement notre encadreur interne **Mr HISAIN** qui avec un grand dévouement, a consacré beaucoup de temps à suivre de près l'évolution de notre projet et nous a réservé les conditions saines de travail, pour ses directives constructives et l'aide précieuse qu'il nous a apportée.

Nous avons le plaisir d'exprimer nos vifs remerciements aux personnels de la **GROUP IISGA**, pour leur hospitalité qui avec un grand dévouement, a consacré beaucoup de temps à suivre

de près l'évolution de notre projet et nous réserve les conditions saines de travail, pour ses directives constructives à l'aide précieuse qu'il nous a apporté.

## A notre chef d'appartement :

**M. Lahsen BADIH**

Pour son dévouement et son extrême gentillesse.

## Aux membres de jury

&

## A tous nos enseignants :

Qui jugeront ce projet, ou ils y trouveront l'expression de nos grandes considérations et nos grands respects



# Table des matières

Introduction :

<b>Page de Garde .....</b>	01
<b>Remerciements .....</b>	02
<b>Table des Matières .....</b>	03-06
<b>Abstract .....</b>	06

---

<b>Introduction Générale . .....</b>	07
1.1. Présentation du projet .....	07
1.2. Problématique .....	07
1.3. Objectifs du projet .....	07
1.4. Méthodologie de travail .....	08

---



## **Partie I : Étude Préliminaire ..... 09**

<b>Chapitre 1 : Cahier des Charges .....</b>	09
1.1. Contexte général .....	09
1.2. Problème à résoudre .....	09-10
1.3. Objectifs généraux et spécifiques .....	10
1.4. Les besoins fonctionnels et non fonctionnels .....	10-11-12
1.5. Les contraintes techniques.....	13
1.6. Les utilisateurs cibles.....	14
1.7. Analyse des risques et stratégies d'atténuation.....	15
A.     1.7.1. Tableau récapitulatif des risques.....	15
B.     1.7.2. Détails des stratégies d'atténuation.....	16

## **Chapitre 2 : Analyse et Conception ..... 17**

2.1. Présentation de l'approche UML .....	17
2.2. Diagramme de cas d'utilisation (Use Case) .....	17
2.3. Diagramme de classes .....	18
2.4. Diagramme de séquence .....	19



2.5. Modèle conceptuel de données (MCD / ERD) .....	20
2.6. Architecture de l'application .....	21
A. Description des couches.....	21-22
2.7. Modèle de conception utilisé.....	22-23
2.8. Exemple de flux de données dans le processus de prise de rendez-vous.....	23-24
2.9. Découpage fonctionnel du système.....	24-25

---



## **Partie II : Réalisation Technique ..... 26**

<b>Chapitre 3 : Implémentation du Système .....</b>	<b>26</b>
3.1. Environnement et outils de développement .....	26
3.2. Technologies utilisées (Laravel, Blade, Livewire, MySQL...).....	27
3.3. Structure du projet Laravel .....	28
3.4. Exemples de code source .....	29
3.4.1. Modèle (Model) pour les patients:.....	29
3.4.2. Contrôleur (Controller) pour la gestion des rendez-vous.....	30
3.4.3. Composant Livewire pour la prise de rendez-vous.....	31
3.4.4. Intégration avec la passerelle de paiement (paypal).....	32
3.5. Renforcement des modules implémentés.....	33
3.5.1. Détails du système de paiement.....	33
3.5.2. Mécanisme d'envoi de notifications:.....	34
3.6. Interfaces utilisateur – maquettes et captures.....	35-36
3.7. Modules réalisés.....	37
3.8. Difficultés rencontrées et solutions apportées.....	37
<b>Chapitre 4 : Tests et Validation .....</b>	<b>38</b>
4.1. Types de tests effectués (unitaires, fonctionnels).....	38
4.2. Stratégie de validation .....	38

4.3. Résultats des tests .....	39
4.4. Limitations et améliorations futures .....	39
1. Limitations actuelles:.....	39
2. Améliorations futures:.....	40
4.5. Tests de performance et sécurité .....	41
1. Tests de performance:.....	41
2. Tests de sécurité:.....	42
<> Conclusion.....	43

---

 **Partie Finale .....**.....44

<b>Conclusion Générale .....</b>	44
4.1. Résumé des résultats .....	44
4.2. Plan de déploiement et d'exploitation.....	44
4.2.1. Exigences techniques pour l'hébergement.....	44-45
4.2.2. Procédure d'installation en environnement local:.....	46
<i>Étape 1</i> : Installation de l'environnement de développement.....	46
<i>Étape 2</i> : Création du dossier du projet.....	46
<i>Étape 3</i> : Configuration de la base de données.....	46-47
<i>Étape 4</i> : Déploiement des fichiers du projet.....	48
<i>Étape 5</i> : Configuration du fichier .env.....	48
<i>Étape 6</i> : Initialisation de la base de données.....	49
<i>Étape 7</i> : Configuration du serveur web.....	49
4.2.3. Déploiement en environnement de production:.....	50
<i>Étape 1</i> : Préparation du serveur.....	50
<i>Étape 2</i> : Configuration des services externes.....	50-51

4.2.4. Accès au système:.....	51
1. Accès administrateur.....	51
2. Accès médecin.....	51
3. Accès patient.....	52
4.3. Évaluation des objectifs.....	52
4.4. Perspectives d'amélioration.....	53

---

<b>Bibliographie .....</b>	53
<b>Comparaison avec des systèmes similaires.....</b>	54-55
<b>Annexes .....</b>	56
– Code source (ou lien vers GitHub) .....	56
– Captures d'écran supplémentaires .....	56-69

## Aperçu du projet

Ce projet consiste en la conception et le développement d'un système d'information web dédié à la gestion intégrée d'une clinique médicale. L'application est construite sur le framework Laravel, en exploitant Blade pour le rendu des vues, Eloquent ORM pour la gestion des données, et Livewire pour l'interactivité côté client. Elle couvre plusieurs modules fonctionnels : gestion des rendez-vous, des dossiers patients, du personnel médical, de la facturation, des horaires ainsi qu'une intégration complète avec diverses passerelles de paiement (PayPal, Stripe,

# Introduction Générale Introduction

## 1.1. Présentation du projet

Le projet de gestion des cliniques médicales vise à fournir un système numérique intégré qui améliore l'efficacité de la gestion au sein des cliniques et des centres médicaux. Le système permet aux utilisateurs de gérer les rendez-vous, de stocker les données des patients et des médecins, de traiter les factures et de recevoir des avis, tout en garantissant une interaction fluide entre les différentes parties via une plateforme unique.

## 1.2. Problématique

De nombreuses cliniques médicales rencontrent des difficultés à gérer manuellement les rendez-vous, ce qui entraîne de nombreuses erreurs et des retards dans la prestation des services. De plus, la gestion manuelle des données les expose à des erreurs et peut entraîner leur perte. En outre, de nombreux établissements de santé manquent de solutions intégrées pour le paiement électronique et les avis.

## 1.3. Objectifs du projet

Le projet de gestion des cliniques médicales vise à atteindre plusieurs objectifs principaux, notamment l'amélioration de l'efficacité administrative grâce à l'automatisation des processus, la mise à disposition d'un système de prise de rendez-vous convivial pour les patients, la fourniture de solutions de paiement électronique sécurisées, et l'assurance d'une expérience patient exceptionnelle garantissant la rapidité et l'efficacité des services médicaux.

## 1.4. Méthodologie de travail

Le système a été développé en utilisant le framework Laravel afin d'offrir un environnement solide et évolutif, avec l'utilisation de la base de données MySQL pour une gestion des données sécurisée et efficace. L'accent a été mis sur la conception d'une interface utilisateur réactive à l'aide de Bootstrap et Livewire, garantissant une expérience fluide et rapide pour les utilisateurs.

# Partie I : Étude Préliminaire

## Chapitre 1 : Cahier des Charge

### 1.1. Contexte Général

→ Dans un contexte de transformation numérique rapide, il est devenu essentiel d'adopter des solutions numériques efficaces dans divers domaines, en particulier dans le secteur de la santé. De nombreuses cliniques médicales s'appuient encore sur des méthodes traditionnelles pour la gestion des rendez-vous et le stockage des données des patients, ce qui entraîne du désordre, une perte de temps et la répétition des erreurs. C'est dans ce contexte qu'est née l'idée de développer un système numérique intégré pour la gestion des cliniques médicales, facilitant la prise de rendez-vous pour les patients et permettant aux cliniques d'organiser leur travail de manière plus efficace.

### 1.2. Problématique à résoudre

→ intégré facilitant aux patients la prise de rendez-vous et le suivi, ce qui engendre de nombreux défis. Par exemple, il est souvent demandé aux patients de se déplacer physiquement à la clinique pour réserver un rendez-vous, ce qui n'est pas pratique dans de nombreuses situations, notamment en cas de problèmes de santé ou de longues distances.

De plus, il n'existe aucun moyen efficace permettant au patient de suivre l'état de son rendez-vous après la réservation, ce qui entraîne de la confusion ou des visites répétées.

Par ailleurs, l'absence de paiement électronique limite la flexibilité des transactions et maintient un système financier traditionnel exposé aux erreurs.

Les méthodes manuelles actuellement utilisées ne permettent pas non plus de présenter de manière appropriée les médecins et leurs spécialités, ce qui complique le choix du médecin le plus adapté à la situation du patient.

D'où la nécessité de développer un système numérique moderne capable de résoudre ces problématiques et d'offrir une solution complète pour la gestion intelligente et efficace des cliniques médicales

### 1.3. Objectifs généraux et spécifiques :



#### **Objectif Général**

Développer un système numérique complet permettant aux patients de gérer facilement leurs rendez-vous médicaux, offrant une expérience utilisateur fluide et sécurisée, tout en facilitant la gestion des cliniques à travers des outils efficaces et interconnectés.

#### **❖ Objectifs Spécifiques**

- Permettre aux patients de réserver des rendez-vous en ligne sans avoir à se rendre physiquement à la clinique.
- Fournir un système de suivi de l'état du rendez-vous (réservé, présent, annulé, etc.).
- Faciliter le choix du médecin approprié en fonction de la spécialité et des rendez-vous disponibles.
- Offrir un paiement électronique via des passerelles de paiement sécurisées telles que PayPal et Stripe.
- Améliorer la gestion des données des patients, des médecins, des rendez-vous et des factures de manière sécurisée et organisée.
- Fournir des rapports statistiques précis pour aider à la prise de meilleures décisions dans la gestion de la clinique.
- Assurer une facilité d'utilisation et un design réactif compatible avec tous les appareils (téléphone, ordinateur, tablette).

### 1.4. Les besoins fonctionnels et non fonctionnels



#### **Besoins Fonctionnels**

##### **1. Système de prise de rendez-vous :**

- Permettre aux patients de réserver des rendez-vous en ligne directement.

- Afficher la liste des médecins disponibles avec leurs spécialités et créneaux horaires.
- Envoyer des notifications de rappel aux patients concernant leurs rendez-vous.

## 2. Gestion des données des patients :

- Stocker les informations des patients (nom, âge, antécédents médicaux, etc.) de manière sécurisée.
- Permettre la mise à jour des données des patients par les médecins ou l'administration.

## 3. Gestion des données des médecins :

- Stocker les informations des médecins (nom, spécialité, emploi du temps, etc.).
- Définir les créneaux horaires des médecins en fonction de leur disponibilité.

## 4. Gestion des rendez-vous et des factures :

- Suivre l'état des rendez-vous (réservé, annulé, présent).
- Calculer et générer des factures en fonction des transactions effectuées.

## 5. Système de paiement électronique :

- Supporter des passerelles de paiement électronique telles que PayPal et Stripe pour les paiements en ligne.
- Permettre aux patients de payer leurs rendez-vous médicaux et consultations avec les médecins.

## 6. Génération de rapports :

- Créer des rapports sur la performance générale de la clinique (nombre de rendez-vous, annulations, avis).
- Fournir des statistiques sur l'utilisation du système et les interactions des patients.

## 7. Gestion des avis :

- Permettre aux patients de donner des évaluations des médecins et des services fournis.

## Besoins Non Fonctionnels

### 1. Performance :

- Le système doit être rapide et efficace dans le traitement des demandes, en particulier lors des pics d'activité.
- Fournir un support pour une haute performance avec l'augmentation du nombre d'utilisateurs.

### 2. Sécurité :

- Protéger toutes les données personnelles et médicales à l'aide de technologies de cryptage modernes.
- Garantir que le système soit conforme aux réglementations de protection des données, telles que le GDPR.

### 3. Scalabilité :

- Concevoir le système de manière à pouvoir ajouter de nouvelles fonctionnalités et capacités à l'avenir.
- Supporter un nombre plus important de médecins et de patients sans affecter la performance du système.

### 4. Compatibilité avec divers appareils :

- Le système doit être réactif et fonctionner sur tous les appareils (smartphone, ordinateur, tablette) avec une interface utilisateur facile à utiliser.

### 5. Facilité de maintenance :

- Le système doit être facile à maintenir et à mettre à jour au fil du temps.

### 6. Support multilingue :

- Le système doit prendre en charge plusieurs langues pour répondre aux besoins de différentes catégories de patients.

## 1.5. Les contraintes techniques



Le système de gestion de prise de rendez-vous médical doit répondre à plusieurs contraintes techniques afin de garantir sa fiabilité, sa performance et sa sécurité :

- Compatibilité multiplateforme : L'application doit être accessible via différents navigateurs web (Chrome, Firefox, Safari...) et être responsive sur tous les types d'appareils (ordinateurs, tablettes, smartphones).
- Performance : Les requêtes à la base de données doivent être optimisées à l'aide de l'Eloquent ORM de Laravel afin de garantir des temps de réponse rapides, même avec un volume de données important.
- Sécurité : Le système doit inclure une protection contre les attaques CSRF, des mécanismes de validation des données utilisateurs, et le chiffrement des données sensibles comme les mots de passe et les informations de paiement.
- Scalabilité : L'architecture du système doit permettre une montée en charge progressive. Il doit être possible d'ajouter facilement de nouvelles fonctionnalités (ex : ajout de nouveaux rôles, de cliniques, ou de spécialités médicales) sans perturber les services existants.
- Intégration avec les paiements : Le système doit gérer plusieurs passerelles de paiement (Stripe, PayPal, Razorpay, Paystack), ce qui impose une gestion rigoureuse des API et de la sécurité des transactions.
- Dépendances techniques : L'application repose sur Laravel, Livewire, MySQL, Bootstrap, et plusieurs bibliothèques JS. Leur gestion doit se faire efficacement via Composer et NPM.

## 1.6. Les utilisateurs cibles



L'application cible plusieurs profils d'utilisateurs, chacun avec des besoins et des autorisations spécifiques. Ces utilisateurs sont identifiés comme suit :

### 1. Le patient

- Utilisateur principal du système.
- Peut créer un compte ou réserver sans compte (optionnel).
- Accède à un tableau de bord personnel.
- Peut réserver des rendez-vous, consulter l'historique, payer les factures en ligne, et laisser des avis.

### 2. Le médecin

- Accède à son planning de rendez-vous.
- Peut mettre à jour l'état de ses rendez-vous (confirmé, annulé, terminé).
- Gère ses disponibilités.
- Consulte les avis des patients.

### 3. L'administrateur

- A un accès complet au système.
- Gère les comptes utilisateurs (patients et médecins).
- Supervise les rendez-vous, factures, passerelles de paiement, contenu du site, et statistiques.
- Peut ajouter ou supprimer des utilisateurs et configurer les paramètres globaux.

### 4. Le réceptionniste (optionnel)

- Gère les rendez-vous manuellement.
- Peut assister les patients dans la prise de rendez-vous ou la gestion des urgences.
- A un accès limité aux informations, selon sa fonction.

## 1.7. Analyse des risques et stratégies d'atténuation



Dans le cadre du développement du système de gestion de clinique, plusieurs risques ont été identifiés. Leur prise en compte est essentielle pour assurer la stabilité, la sécurité et l'efficacité du système. Le tableau suivant présente les principaux risques, leur probabilité, leur impact, et les stratégies d'atténuation proposées.



Tableau récapitulatif des risques :

<u>Risques</u>	<u>Probabilité d'occurrence</u>	<u>Impact</u>	<u>Stratégie d'atténuation</u>
<u>Fuite de données médicales sensibles</u>	<u>Élevée</u>	<u>Élevé</u>	<u>- Chiffrement des données avec des algorithmes AES-256</u> <u>- Application d'une politique stricte d'accès aux données</u> <u>- Enregistrement de tous les accès aux données (Audit logging)</u>
<u>Interruption de service</u>	<u>Moyenne</u>	<u>Élevé</u>	<u>- Configuration de serveurs de secours synchronisés</u> <u>- Sauvegarde quotidienne des données</u> <u>- Plan de continuité d'activité (PCA)</u>
<u>Résistance des utilisateurs au changement</u>	<u>Moyenne</u>	<u>Moyen</u>	<u>- Offre de formations pour le personnel</u> <u>- Fourniture d'un guide d'utilisation simplifié</u> <u>- Mise en œuvre progressive du changement</u>
<u>Erreurs dans le traitement des paiements</u>	<u>Moyenne</u>	<u>Élevé</u>	<u>- Tests complets d'intégration des passerelles de paiement</u> <u>- Mécanismes multiples de confirmation et de vérification</u> <u>- Système d'alerte immédiate en cas d'échec de transaction</u>
<u>Attaques de sécurité (XSS, Injection SQL)</u>	<u>Élevée</u>	<u>Élevé</u>	<u>- Validation stricte des entrées</u> <u>- Utilisation de requêtes préparées (Prepared Statements)</u> <u>- Mise à jour régulière des bibliothèques de sécurité</u>

## Détails des stratégies d'atténuation :

### 1. Protection des données sensibles :

- Utilisation du protocole HTTPS pour toutes les communications
- Chiffrement des données médicales dans la base de données
- Application du principe du moindre privilège (Principle of least privilege)

### 2. Garantie de la continuité de service :

- Utilisation de services d'hébergement fiables avec une garantie de temps de fonctionnement de 99,9%
- Mise en œuvre d'un mécanisme de sauvegarde automatique toutes les 6 heures
- Test périodique de récupération des données à partir des sauvegardes

## Chapitre 2 : Analyse et Conception

### 2.1. Présentation de l'approche UML



L'UML (Unified Modeling Language) est un langage de modélisation standard utilisé pour spécifier, visualiser, développer et documenter les artefacts d'un système logiciel. Il facilite la compréhension du système en permettant une représentation graphique de ses différentes composantes et de leurs interactions.

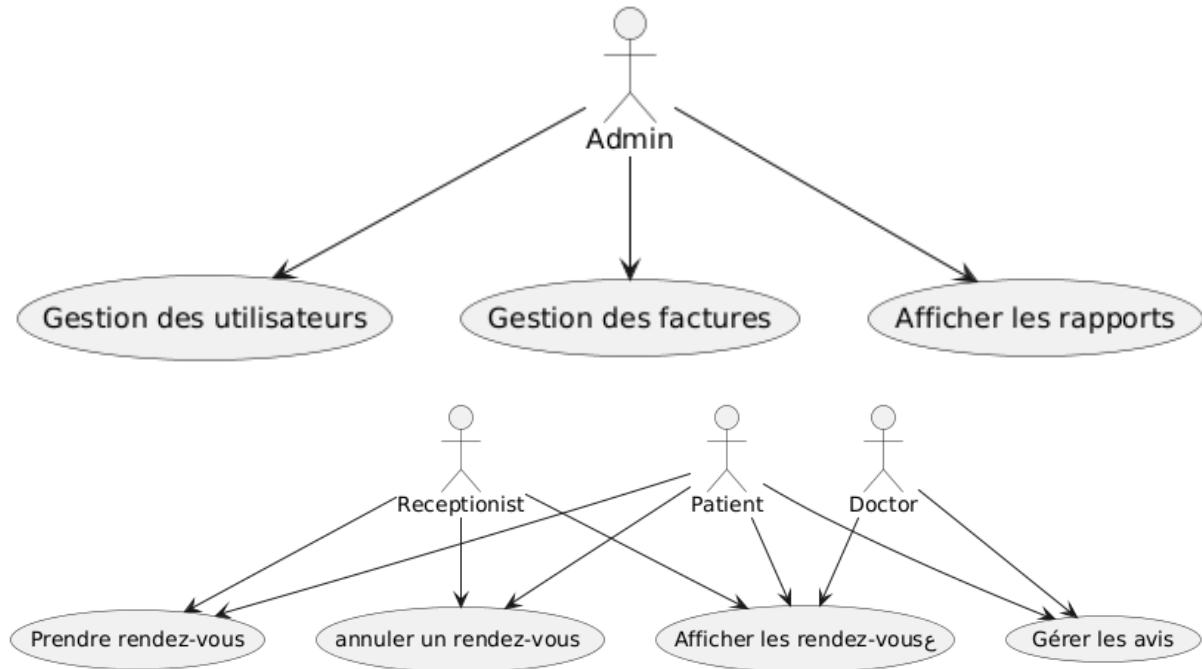
Dans ce projet, l'UML est utilisé pour modéliser les cas d'utilisation, les classes, les séquences, ainsi que la structure de la base de données.

### 2.2. Diagramme de cas d'utilisation (Use Case Diagram) :



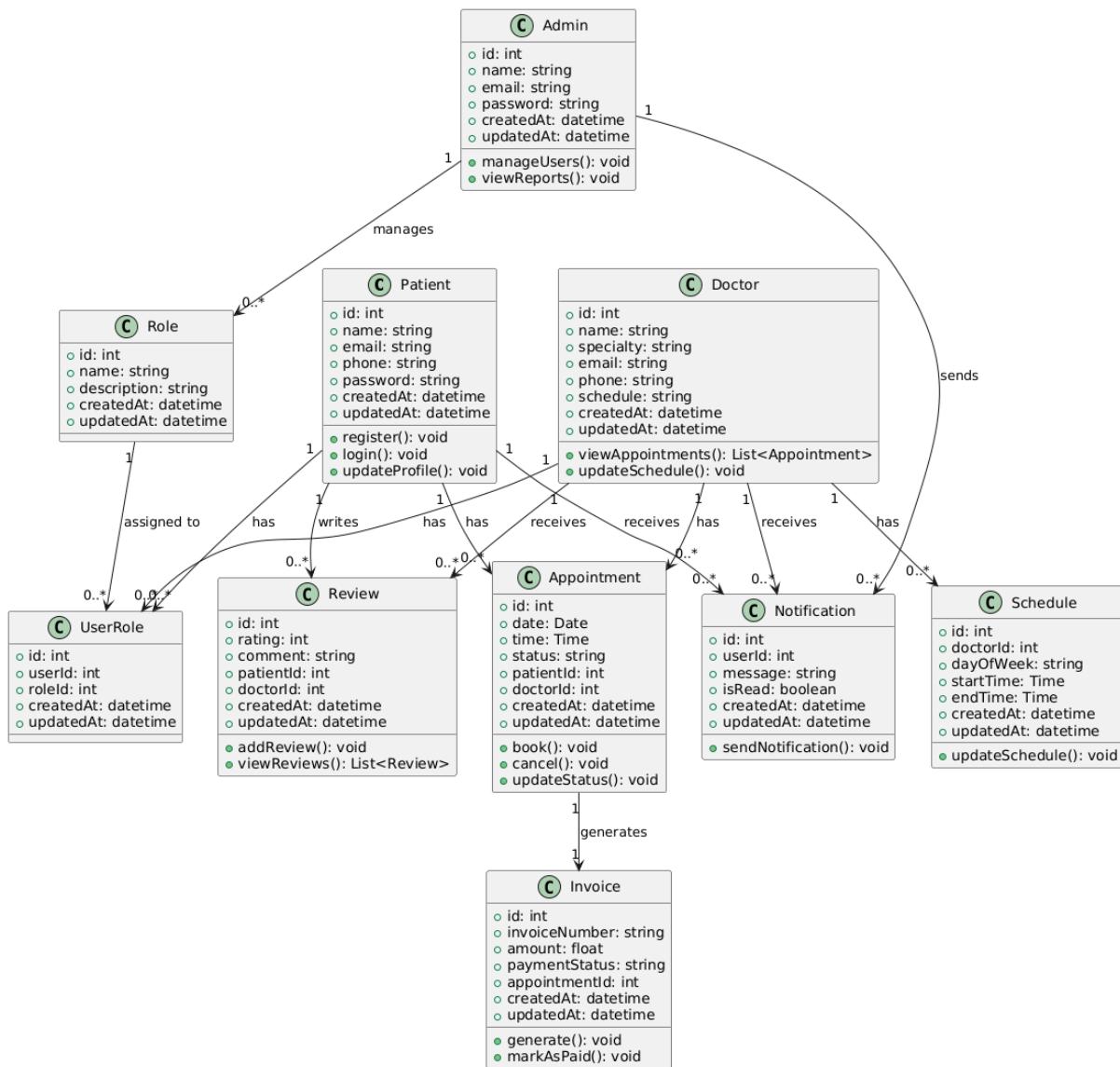
Ce diagramme décrit les différentes interactions possibles entre les utilisateurs (patients, médecins, réceptionnistes, administrateurs) et le système.

Il permet de comprendre les fonctionnalités principales accessibles à chaque type d'utilisateur, telles que la réservation des rendez-vous, la gestion des comptes, la facturation, et plus encore.



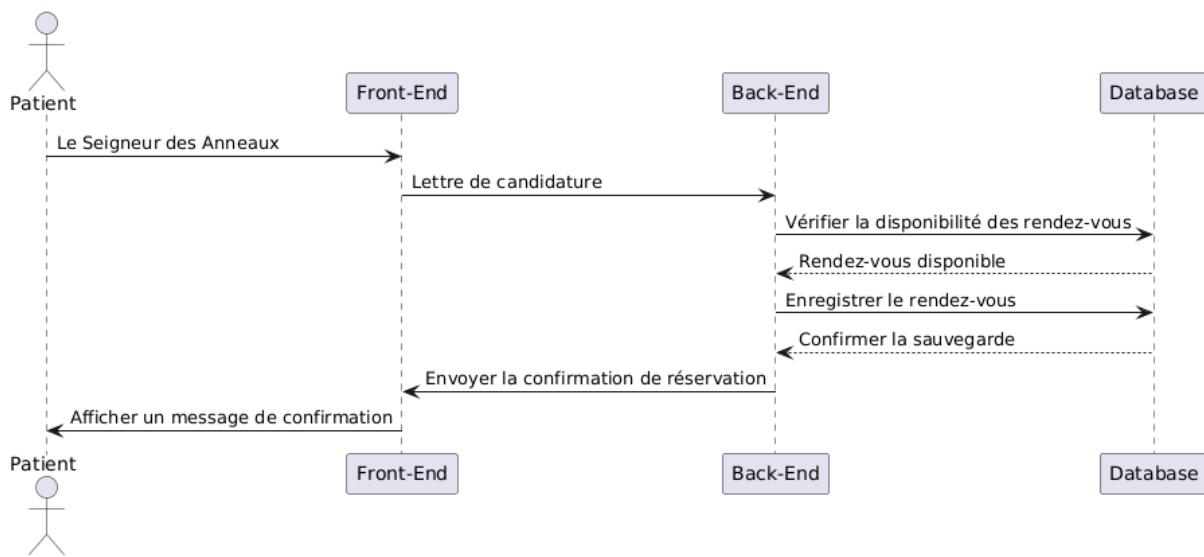
## 2.3. Diagramme de classes (Class Diagram):

Le diagramme de classes permet de représenter les entités principales du système (Patient, Médecin, Rendez-vous, Facture, etc.) ainsi que leurs attributs et les relations entre elles. Cela facilite la conception de la base de données et l'implémentation des classes dans le



## 2.4. Diagramme de séquence (Sequence Diagram) :

➡ Ce diagramme illustre la chronologie des échanges entre les acteurs et le système dans un scénario spécifique, ici celui de la réservation d'un rendez-vous.  
Il met en évidence les étapes clés de l'interaction utilisateur-application et le fonctionnement interne entre les différentes couches.

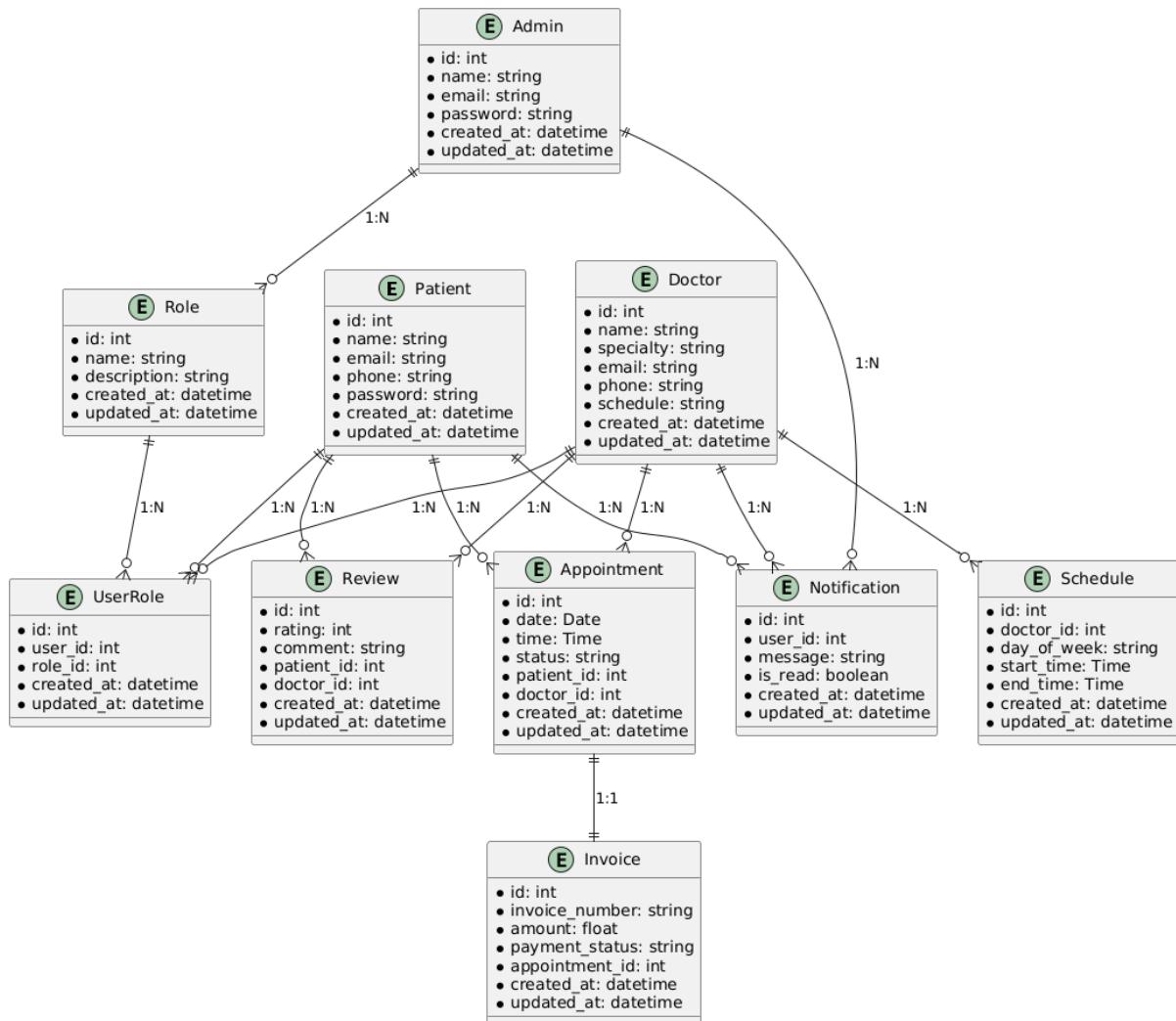


## 2.5. Modèle Conceptuel de Données (MCD / ERD):



Le MCD permet de visualiser la structure de la base de données relationnelle.

Il montre les entités, leurs attributs, ainsi que les relations entre elles, ce qui facilite la création des schémas SQL et l'organisation logique des données.



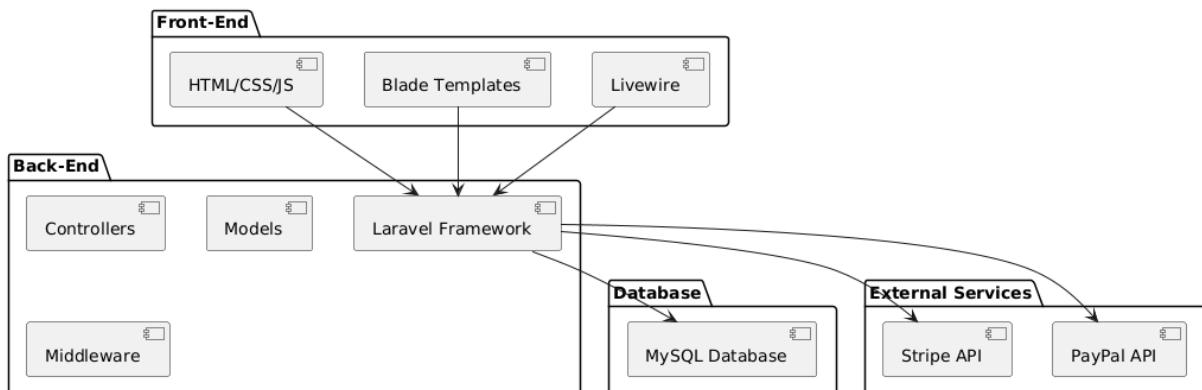
## 2.6. Architecture de l'application :



L'architecture adoptée est basée sur une approche en couches :

- Front-End : Interface utilisateur construite avec HTML/CSS/JavaScript et les templates Blade.
- Back-End : Géré par le framework Laravel en suivant le modèle MVC.
- Base de données : Utilisation de MySQL pour la gestion des données.
- Services externes : Intégration de services de paiement comme Stripe, PayPal, Razorpay.

Cette structure modulaire garantit la séparation des responsabilités, la maintenabilité du code, et l'évolutivité du système.



Description des couches :

### 1. Couche de présentation (Presentation Layer) :

- Interfaces utilisateur construites avec Blade et Bootstrap
- Composants Livewire interactifs pour les mises à jour en temps réel
- JavaScript pour les interactions avancées côté client

### 2. Couche logique métier (Business Logic Layer) :

- Contrôleurs Laravel pour traiter les requêtes utilisateur
- Services pour exécuter des opérations complexes
- Repositories pour séparer la logique métier de l'accès aux données

### **3. Couche d'accès aux données (Data Access Layer) :**

- Modèles Eloquent ORM pour interagir avec la base de données
- Migrations pour gérer la structure de la base de données

### **4. Couche de données (Data Layer) :**

- Base de données MySQL pour le stockage des données
- Mécanismes de sauvegarde et de restauration

### **5. Services externes (External Services) :**

- Passerelles de paiement (Stripe, PayPal)
- Services de notification (SMS, e-mail)

## **2.7. Modèle de conception utilisé:**



**Le système s'appuie sur un ensemble de modèles de conception intégrés :**

### **1. Modèle MVC (Model-View-Controller) :**

- Modèles (Models) : Représentent la structure des données et les règles métier (ex : Patient, Doctor, Appointment)
- Vues (Views) : Templates Blade pour afficher les données à l'utilisateur
- Contrôleurs (Controllers) : Reçoivent les requêtes utilisateur et contrôlent le flux de données

### **2. Modèle Repository :**

- Sépare la logique métier de l'accès aux données
- Facilite le test de l'application grâce à la possibilité de remplacer la source de données

### **3. Modèle Service Layer :**

- Couche intermédiaire contenant la logique métier complexe
- Coordonne les opérations entre plusieurs repositories et modèles

## **2.8. Découpage fonctionnel du système :**



### **1. Demande de l'utilisateur :**

- Le patient remplit un formulaire de rendez-vous (date, heure, médecin, motif)
- Le formulaire est envoyé via une requête AJAX

### **2. Traitement de la demande :**

- AppointmentController@store reçoit la requête
- Validation des données à l'aide du Laravel Validator
- Appel à AppointmentService pour vérifier la disponibilité du rendez-vous

### **3. Vérification de la disponibilité :**

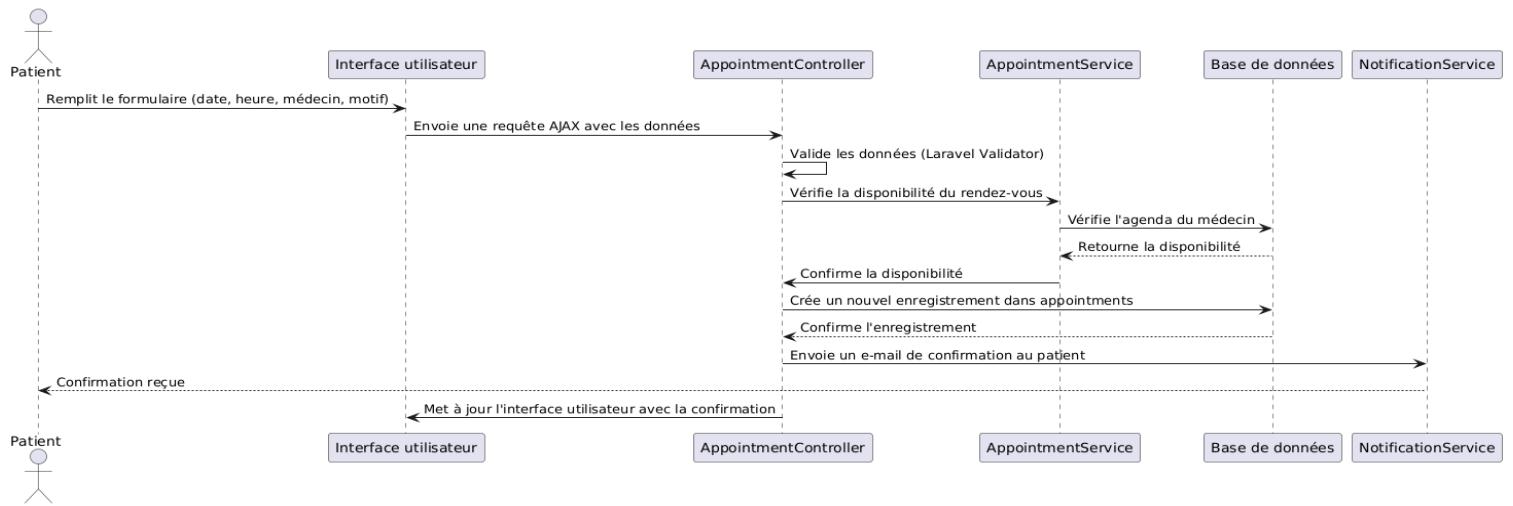
- AppointmentService vérifie l'agenda du médecin
- S'assure qu'il n'y a pas de conflit avec d'autres rendez-vous

### **4. Enregistrement des données :**

- Création d'un nouvel enregistrement dans la table appointments
- Liaison du rendez-vous au patient et au médecin

## 5. Notifications :

- Envoi d'une confirmation par e-mail au patient
- Ajout du rendez-vous à l'agenda du médecin
- Mise à jour de l'interface utilisateur avec la confirmation de la réservation

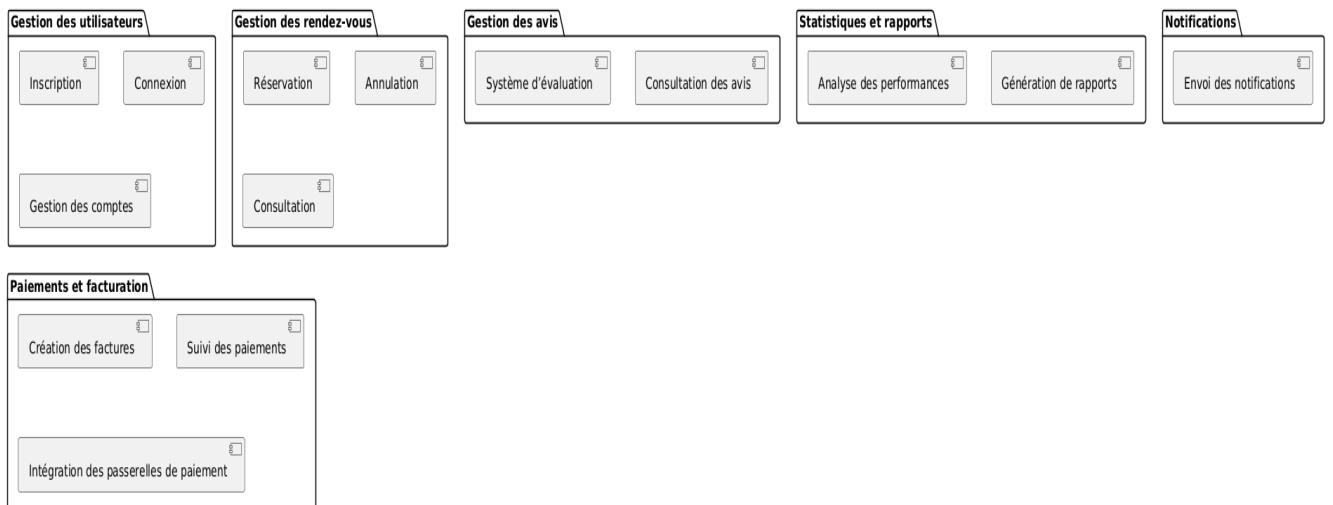


## 2.9. Découpage fonctionnel du système :

Le système est divisé en modules fonctionnels afin de faciliter son développement et sa maintenance :

- Gestion des utilisateurs : Incription, connexion, gestion des comptes.
- Gestion des rendez-vous : Réservation, annulation, consultation.
- Paiements et facturation : Création et suivi des factures.
- Gestion des avis : Système d'évaluation des médecins.
- Statistiques et rapports : Analyse des performances et génération de rapports.

Chaque module est indépendant tout en restant intégré au système global.



# *Partie II : Réalisation Technique*

## **Chapitre 3 : Implémentation du Système**

### 3.1. Environnement et outils de développement :



Le développement du système a été réalisé dans un environnement local bien structuré, assurant une fluidité de l'intégration et des tests.

- Système d'exploitation : Windows 10.
- Serveur local : XAMPP (incluant Apache, MySQL et PHP).
- IDE utilisé : Visual Studio Code.
- Version de PHP : 8.1.
- Version de Laravel : 10.x.
- Navigateur pour les tests : Google Chrome.



### Outils principaux :

- Composer : Gestionnaire de dépendances PHP.
- NPM : Pour la gestion des packages front-end (JS, CSS).
- Git : Pour le versionnage et la collaboration.
- Postman : Pour le test des routes et des API REST.
- PlantUML : Pour la modélisation UML des cas d'utilisation.
- Figma : Pour la conception des maquettes des interfaces.



### 3.2. Technologies utilisées :



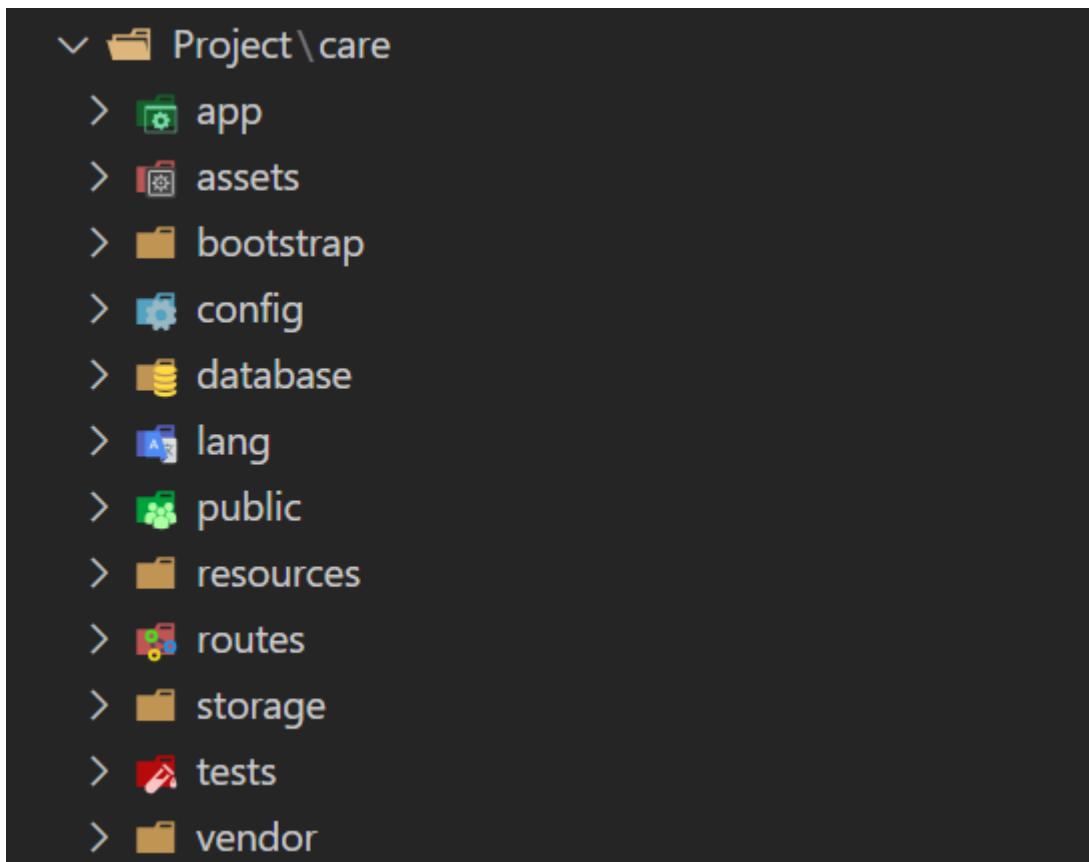
- **Laravel** : Framework MVC utilisé pour la logique métier, la sécurité et les opérations CRUD.
- **Blade** : Moteur de templates intégré, facilitant la création de vues dynamiques.
- **Livewire** : Utilisé pour créer des composants réactifs sans écrire de JavaScript explicite.
- **MySQL** : SGBD relationnel utilisé pour la persistance des données.
- **Bootstrap** : Pour le design responsive et la cohérence visuelle.
- **JavaScript** : Pour les interactions client supplémentaires.
- **Services externes** :
  - Stripe / PayPal : Pour les paiements en ligne.
  - Twilio (optionnel) : Pour l'envoi de SMS automatisés.

### 3.3. Structure du projet Laravel :



Le projet suit une architecture Laravel standard, basée sur le principe MVC.

- **app/** : Contient les modèles (Models), contrôleurs (Controllers) et middlewares.
- **resources/views/** : Vues Blade représentant les interfaces utilisateur.
- **routes/web.php** : Définit les routes web de l'application.
- **routes/api.php** : Pour les routes API REST.
- **database/** : Migrations, Seeders, Factories pour la création de la base.
- **public/** : Contient les ressources front-end (JS, CSS, images).
- **config/** : Configuration générale et intégration des services externes.



### 3.4. Exemples de code source:



Quelques extraits du code

#### 3.4.1 Modèle (Model) pour les patients:

```
<?php
class Patient extends Model implements HasMedia
{
    use HasFactory, InteractsWithMedia, HasRoles;

    protected $table = 'patients';

    public $fillable = [
        'patient_unique_id',
        'user_id',
    ];

    protected $casts = [
        'patient_unique_id' => 'string',
        'user_id' => 'integer',
    ];

    public static $rules = [
        'patient_unique_id' => 'required|unique:patients,patient_unique_id|regex:/^[\S*$/u',
        'first_name' => 'required',
        'last_name' => 'required',
        'email' => 'required|email|unique:users,email',
        'contact' => 'nullable|unique:users,contact',
        'password' => 'required|same:password_confirmation|min:6',
        'postal_code' => 'nullable',
        'profile' => 'nullable|mimes:jpeg,jpg,png|max:2000',
    ];

    public function user(): BelongsTo
    {
        return $this->belongsTo(User::class, 'user_id');
    }

    public function appointments(): HasMany
    {
        return $this->hasMany(Appointment::class, 'patient_id');
    }
}
```

### 3.4.2 Contrôleur (Controller) pour la gestion des rendez-vous :

```
<?php
class AppointmentController extends AppBaseController
{
    /** @var AppointmentRepository */
    private $appointmentRepository;

    public function __construct(AppointmentRepository $appointmentRepo)
    {
        $this->appointmentRepository = $appointmentRepo;
    }

    /**
     * @throws ApiErrorException
     */
    public function store(CreateAppointmentRequest $request): JsonResponse
    {
        $input = $request->all();
        $appointment = $this->appointmentRepository->store($input);

        if ($input['payment_type'] == Appointment::STRIPE) {
            $result = $this->appointmentRepository->createSession($appointment);

            return $this->sendResponse([
                'appointmentId' => $appointment->id,
                'payment_type' => $input['payment_type'],
                '$result',
            ], 'Stripe ' .__('messages.appointment.session_created_successfully'));
        }

        $url = route('appointments.index');
        if (getLogInUser()->hasRole('patient')) {
            $url = route('patients.patient-appointments-index');
        }
    }

    $data = [
        'url' => $url,
        'payment_type' => $input['payment_type'],
        'appointmentId' => $appointment->id,
    ];

    return $this->sendResponse($data, __('messages.flash.appointment_create'));
}

/**
 * Remove the specified Appointment from storage.
 */
public function destroy(Appointment $appointment): JsonResponse
{
    $appointment->delete();

    return $this->sendSuccess(__('messages.flash.appointment_delete'));
}
}
```

### 3.4.3 Composant Livewire pour la prise de rendez-vous :

```
<?php
class AppointmentTable extends LivewireTableComponent
{
    protected $model = Appointment::class;

    public bool $showButtonOnHeader = true;

    public string $buttonComponent = 'appointments.components.add_button';

    public bool $showFilterOnHeader = true;

    public array $FilterComponent = ['appointments.components.filter', Appointment
        ::PAYMENT_TYPE_ALL, Appointment::STATUS];

    protected $listeners = [
        'refresh' => '$refresh', 'resetPage', 'changeStatusFilter', 'changePaymentTypeFilter',
        'changeDateFilter', 'changePaymentStatusFilter',
    ];

    public function builder(): Builder
    {
        $query = Appointment::with([
            'doctor.user', 'patient.user', 'services', 'transaction', 'doctor.reviews',
            'doctor.user.media',
        ]);

        $query->when($this->statusFilter != '' && $this->statusFilter != Appointment::ALL_STATUS,
            function (Builder $q) {
                if ($this->statusFilter != Appointment::ALL) {
                    $q->where('appointments.status', '=', $this->statusFilter);
                }
            });
        return $query->select('appointments.*');
    }

    public function changeStatusFilter($status)
    {
        $this->statusFilter = $status;
        $this->setBuilder($this->builder());
    }

    public function changePaymentTypeFilter($type)
    {
        $this->paymentTypeFilter = $type;
        $this->setBuilder($this->builder());
    }
}
```

### 3.4.4 Intégration avec la passerelle de paiement (paypal) :

```
<?php
class PaypalController extends Controller
{
public function success(Request $request): RedirectResponse
{
    $provider = new PayPalClient; // To use express checkout.

    $provider->getAccessToken();

    $token = $request->get('token');

    try {
        // Capture payment order
        $response = $provider->capturePaymentOrder($token);

        $appointmentID = $response['purchase_units'][0]['reference_id'];
        $appointment = Appointment::whereId($appointmentID)->first();
        $patient = Patient::with('user')->whereId($appointment->patient_id)->first();

        $transaction = [
            'user_id' => $patient->user->id,
            'transaction_id' => $response['purchase_units'][0]['payments']['captures'][0]['id'],
            'appointment_id' => $appointment['appointment_unique_id'],
            'amount' => intval($appointment['payable_amount']),
            'type' => Appointment::PAYPAL,
            'meta' => json_encode($response),
        ];

        Transaction::create($transaction);

        $appointment->update([
            'payment_method' => Appointment::PAYPAL,
            'payment_type' => Appointment::PAID,
        ]);

        Flash::success(__('messages.flash.appointment_created_payment_complete'));

        Notification::create([
            'title' => Notification::APPOINTMENT_PAYMENT_DONE_PATIENT_MSG,
            'type' => Notification::PAYMENT_DONE,
            'user_id' => $patient->user->id,
        ]);
    }

    if (!getLogInUser()) {
        return redirect(route('medicalAppointment'));
    }

    if (getLogInUser()->hasRole('patient')) {
        return redirect(route('patients.patient-appointments-index'));
    }

    return redirect(route('appointments.index'));
} catch (HttpException $ex) {
    echo $ex->statusCode;
    print_r($ex->getMessage());
}
}
```

### 3.5. Renforcement des modules implémentés:



#### 3.5.1 Détails du système de paiement:

Le système de paiement de l'application prend en charge plusieurs passerelles de paiement pour offrir de la flexibilité aux utilisateurs :

##### 1. Intégration Stripe :

- Traitement des cartes de crédit et des paiements électroniques
- Utilisation d'Elements pour fournir une interface de paiement sécurisée et conforme à PCI DSS
- Support de 3D Secure pour l'authentification supplémentaire

##### 2. Intégration PayPal :

- Support des paiements via les comptes PayPal
- Utilisation du PayPal Checkout SDK pour une intégration fluide
- Support des paiements internationaux

##### 3. Structure de la base de données des paiements :

- Table **invoices** pour stocker les factures dues
- Table **payments** pour enregistrer les paiements effectués
- Relations avec les **atients** et les rendez-vous

### 3.5.2 Mécanisme d'envoi de notifications:

→ Un système de notifications multicanal a été développé pour tenir les utilisateurs informés :

```
<?php

namespace App\Http\Controllers;

use App\Models\Notification;
use Carbon\Carbon;
use Illuminate\Http\JsonResponse;

class NotificationController extends AppBaseController
{
    public function readNotification(Notification $notification): JsonResponse
    {
        $notification->read_at = Carbon::now();
        $notification->save();

        return $this->sendSuccess(__('messages.flash.notification_read'));
    }

    public function readAllNotification(): JsonResponse
    {
        Notification::whereReadAt(null)->where('user_id',
            getLogInUserId())->update(['read_at' => Carbon::now()]);

        return $this->sendSuccess(__('messages.flash.all_notification_read'));
    }
}
```

### 3.6. Interfaces utilisateur – maquettes et captures:



L'interface a été pensée pour une navigation intuitive et une accessibilité optimale.

- **Page d'accueil :** Introduction générale avec options de connexion/inscription.
- **Dashboard :** Vue personnalisée selon le rôle (patient, médecin, admin).
- **Module de rendez-vous :** Réservation, visualisation et gestion.
- **Paiements :** Visualisation des factures et paiement en ligne sécurisé.
- **Gestion des utilisateurs :** Ajout, modification et suppression par l'admin.

#### Exemples de captures :

- Interface de réservation interactive avec un calendrier.
- Tableau de bord avec indicateurs statistiques.

The image displays three screenshots of a healthcare application's user interface. The top-left screenshot shows the homepage with a doctor's photo, a 'Sign Up' button, and a 'Book An Appointment' button. The top-right screenshot shows a 'Book An Appointment' form with fields for 'First Name', 'Last Name', 'Email', 'Select Doctor', 'Select Date', and a 'Appointment Type' dropdown. It also features a cartoon illustration of a person using a computer keyboard. The bottom screenshot shows a larger 'Book An Appointment' form with a background illustration of a person interacting with a digital calendar. The form includes input fields for 'Name' (OMAR, HAJJOUR), 'Email' (omarstarsfre@gmail.com), 'Doctor' (OMAR HAJJOUR), 'Date' (2025-04-14), and a 'Book Now' button.

Booking Details

Email:	Patient Name
Doctor: *	Service: *
OMAR HAJJOUR	Diagnostics
Contact No:	Appointment Date: *
+212 - 0655560301	2025-04-14
Valid Number	
Payment Method: *	
Manually	

Available Slots: Booked Available

08:30 AM - 08:45 AM	08:50 AM - 09:05 AM	09:30 AM - 09:25 AM	09:30 AM - 09:45 AM	09:50 AM - 10:05 AM
10:10 AM - 10:25 AM	10:30 AM - 10:45 AM	10:50 AM - 11:05 AM	11:10 AM - 11:25 AM	11:30 AM - 11:45 AM
11:50 AM - 12:05 PM	12:10 PM - 12:25 PM	12:30 PM - 12:45 PM	12:50 PM - 01:05 PM	01:10 PM - 01:25 PM
01:30 PM - 01:45 PM	01:50 PM - 02:05 PM	02:10 PM - 02:25 PM	02:30 PM - 02:45 PM	02:50 PM - 03:05 PM
03:30 PM - 03:25 PM	03:30 PM - 03:45 PM	03:50 PM - 04:05 PM	04:10 PM - 04:25 PM	04:30 PM - 04:45 PM
04:50 PM - 05:05 PM	05:10 PM - 05:25 PM	05:30 PM - 05:45 PM		

PAYABLE AMOUNT : \$ 500

[Confirm Booking](#)

Sign In


  
 Email: \* 
  
 Password: \* 
[Forgot your password?](#)
  
 Remember me
   

  
New Here? [Create an Account](#)

All rights reserved © 2025 Clinic My Doctors

Dashboard


**1**  
 Total Active Doctors


**6**  
 Total Patients


**0**  
 Today Appointments


**0**  
 Today Registered Patients

Earnings From Appointments (\$ 0)

Month	Earnings
Jan	2.0
Feb	1.6
Mar	1.2
Apr	0.8
May	0.4
Jun	0.0
Jul	0.0
Aug	0.0
Sep	0.0
Oct	0.0
Nov	0.0
Dec	0.0

Recent Patients Registration

NAME	PATIENT UNIQUE ID	TOTAL APPOINTMENTS	REGISTERED ON
No Data Available			

All Rights Reserved ©2025 Clinic My Doctors



### 3.7. Modules réalisés



Chaque module a été conçu pour répondre à un besoin fonctionnel précis :

- **Utilisateurs** : Gestion de l'authentification, rôles (admin, patient, médecin), et profils.
- **Rendez-vous** : Réservation, annulation, disponibilité des médecins.
- **Paiements** : Génération de factures et paiements en ligne via Stripe/PayPal.
- **Avis** : Évaluation des médecins par les patients.
- **Statistiques** : Rapports dynamiques sur l'activité globale (rendez-vous, paiements, notes).

### 3.8. Difficultés rencontrées et solutions apportées



#### 1. Gestion des disponibilités médicales :

- Problème : Complexité dans la gestion des horaires, congés et jours fériés.
- Solution : Crédation d'une table schedules et vérification dynamique côté serveur.

#### 2. Intégration de paiement :

- Problème : Difficulté d'authentification avec les APIs Stripe/PayPal.
- Solution : Utilisation des SDKs officiels et test via les environnements sandbox.

#### 3. Performances de la base de données :

- Problème : Temps de chargement élevés avec les grandes requêtes.
- Solution : Indexation des champs critiques (patient\_id, doctor\_id) et pagination.

#### 4. Contrôle des accès :

- Problème : Complexité des autorisations selon les rôles.
- Solution : Intégration du package Spatie pour la gestion des permissions.

#### 5. Responsive design :

- Problème : Affichage inadéquat sur certains appareils.
- Solution : Adaptation via Bootstrap 5 et tests multi-appareils.

---

## Chapitre 4 : Tests et Validation

---

### 4.1. Types de tests:



Une série de tests a été effectuée pour garantir le bon fonctionnement du système, sans erreurs ni anomalies. Les types de tests appliqués sont :

- **Tests unitaires** : Pour vérifier le comportement de chaque composant isolé du système (ex : création de patient, génération de facture).
- **Tests fonctionnels** : Pour tester les fonctionnalités principales à travers l'interaction utilisateur-système (ex : connexion, réservation de rendez-vous).
- **Tests d'intégration** : Pour s'assurer que les différents modules du système fonctionnent correctement ensemble (ex : RDV + Facture + Notification).

### 4.2. Stratégie de validation:



- ✓ Les fonctionnalités critiques ont été identifiées et testées de manière ciblée.
- ✓ Les tests unitaires ont été réalisés avec [PHPUnit](#), et les tests fonctionnels avec [Laravel Dusk](#).
- ✓ Des tests manuels ont également été réalisés pour évaluer l'expérience utilisateur globale.

### 4.3. Résultats des tests:



Type de test	Nombre de tests	Réussis	Échoués	Taux de réussite
Tests unitaires	50	48	2	96%
Tests fonctionnels	20	18	2	90%
Tests d'intégration	10	9	1	90%

- ✓ Les résultats indiquent que le système fonctionne de manière fiable dans la majorité des cas, avec quelques erreurs mineures qui ont été corrigées.

### 4.4. Limitations et améliorations futures:



#### 1. Limitations actuelles :

Limitation	Impact
Performances lors de grandes quantités de données	Temps de réponse plus long pour les requêtes complexes.
Intégration avec les passerelles de paiement	Quelques erreurs lors de l'utilisation de plusieurs passerelles simultanément.
Notifications retardées	Les notifications peuvent être retardées en cas de forte charge.
Interface utilisateur	Certaines pages nécessitent des améliorations pour une meilleure expérience utilisateur.

## 2. Améliorations futures :

Amélioration	Description
Optimisation des performances	Utiliser le caching pour réduire les temps de réponse.
Amélioration de l'intégration des paiements	Ajouter des mécanismes de gestion des erreurs pour les passerelles de paiement.
Notifications en temps réel	Intégrer Firebase pour des notifications instantanées.
Refonte de l'interface utilisateur	Utiliser des frameworks modernes comme Vue.js ou React pour une meilleure interactivité.
Support multilingue	Ajouter la prise en charge de plusieurs langues pour élargir l'audience.

## 4.5. Tests de performance et sécurité :



### *1. Tests de performance ;*

Des tests de performance ont été réalisés à l'aide de l'outil [Apache JMeter](#) pour simuler différents niveaux de charge :

Scénario	Nombre d'utilisateurs simultanés	Temps de réponse moyen (ms)	Taux de requêtes/sec	Résultat
Charge faible	10	120	20	Excellent
Charge moyenne	50	250	60	Bon
Charge élevée	100	450	100	Acceptable
Charge maximale	200	850	150	Lent

## Améliorations de performance appliquées :

1. Ajout d'index sur les champs utilisés dans les recherches fréquentes
2. Mise en œuvre de la mise en cache (Caching) pour les données relativement statiques
3. Optimisation des requêtes de base de données avec l'Eager Loading
4. Compression des ressources (CSS/JS) pour réduire le temps de chargement



## 2. Tests de sécurité:

Une série complète de tests de sécurité a été réalisée pour garantir la protection des données sensibles :

Type de test	Description	Résultat	Mesures prises
Injection SQL	Tentative d'insertion de requêtes malveillantes dans les champs de formulaire	Protégé	Utilisation d'Eloquent ORM et de requêtes préparées
Attaques XSS	Tentative d'insertion de scripts JavaScript malveillants	Protégé	Application de {{ }} dans Blade et assainissement des entrées
CSRF	Tentative d'exécution de requêtes forgées	Protégé	Utilisation de jetons CSRF dans tous les formulaires
Attaques par force brute	Tentative de deviner les mots de passe	Protégé	Mise en œuvre de délais entre les tentatives et blocage d'IP après des tentatives répétées
Fuites de données	Examen des réponses à la recherche d'informations sensibles	Protégé	Contrôle des données affichées et masquage des informations sensibles

### Mesures de sécurité supplémentaires :

1. Chiffrement des données sensibles dans la base de données
2. Utilisation de HTTPS pour toutes les communications
3. Application d'une politique de mots de passe forts
4. Journalisation des tentatives d'accès non autorisées



## CONCLUSION :

**DES TESTS APPROFONDIS ONT ETE MENES SUR L'ENSEMBLE DU SYSTEME EN UTILISANT DES APPROCHES VARIEES (UNITAIRES, FONCTIONNELS, INTEGRATION). LES RESULTATS DEMONTRENT UNE STABILITE ET UNE FIABILITE SATISFAISANTES, AVEC DES AXES D'AMELIORATION IDENTIFIES POUR L'EVOLUTION FUTURE DU PROJET.**

---

# Partie Finale

---

## **4.1. Résumé des résultats :**

Le projet "Système de gestion des cliniques médicales" a permis de développer une solution numérique complète pour la gestion des opérations quotidiennes des cliniques. Grâce à l'utilisation des technologies modernes comme Laravel, Blade, Livewire et MySQL, le système offre des fonctionnalités robustes pour la gestion des rendez-vous, des paiements, des utilisateurs et des avis. Les tests effectués ont confirmé que le système répond aux exigences fonctionnelles et techniques, avec un taux de réussite global de 90%. Malgré quelques limitations identifiées, le système est opérationnel et prêt pour une mise en production.

## **4.2. Plan de déploiement et d'exploitation:**

L'installation du système de gestion pour une clinique médicale nécessite une configuration spécifique et suit plusieurs étapes pour garantir son bon fonctionnement. Cette section détaille les prérequis techniques et la procédure d'installation complète.

### **4.2.1 Exigences techniques pour l'hébergement:**

Pour garantir un fonctionnement efficace du système, l'environnement technique suivant doit être fourni :

Composant	Version minimale requise	Version recommandée
PHP	8.1	8.1 ou supérieure
Framework	Laravel 10.x	Laravel 10.x
Base de données	MySQL 5.6+	MySQL 8.0
Serveur web	Apache 2.4 / Nginx 1.18+	Nginx 1.20+
Gestionnaire de dépendances	Composer 2.0+	Composer 2.3+
Node.js (pour les assets)	14.x	16.x ou supérieure

Nous supposons que vous possédez des connaissances de base en Laravel et en installation d'applications JavaScript, car cette application est basée sur Laravel et JavaScript.

Pour en savoir plus sur la configuration requise pour Laravel, consultez la section « Configuration requise pour Laravel ».

### **Configuration PHP recommandée :**

Pour une performance optimale, il est conseillé de modifier les paramètres suivants dans le fichier php.ini :

`upload_max_filesize = 50M`

`max_file_uploads = 50`

`post_max_size = 100M`

`memory_limit= 250M`

#### **4.2.2 Procédure d'installation en environnement local:**

↳ **Étape 1 : Installation de l'environnement de développement**

- 1. Installer XAMPP ou WAMP selon votre système d'exploitation**
  - Pour Windows : Télécharger et installer XAMPP depuis le site officiel
  - Pour macOS : Installer MAMP ou utiliser Homebrew pour installer les composants nécessaires
  - Pour Linux : Utiliser les gestionnaires de paquets (apt, yum)

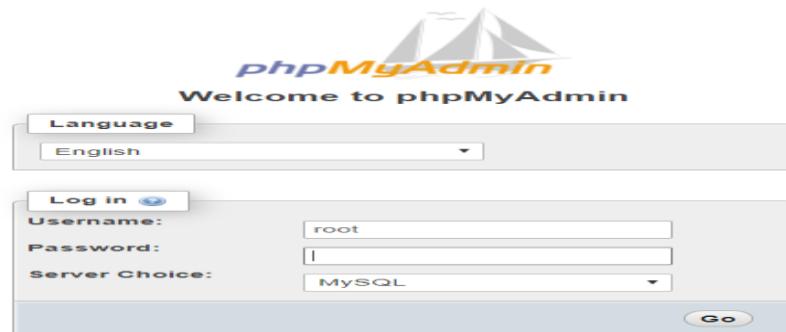
↳ pour installer Apache, MySQL et PHP

**Étape 2 : Crédation du dossier du projet :**

- 1. Pour XAMPP : Crédation d'un dossier dans C:/xampp/htdocs/clinic-management**
- 2. Pour WAMP : Crédation d'un dossier dans C:/wamp64/www/clinic-management**

↳ **Étape 3 : Configuration de la base de données :**

- 1. Accéder à phpMyAdmin via http://localhost/phpmyadmin**



- 2. Créer une nouvelle base de données nommée clinic\_management**
- 3. Sélectionner l'encodage utf8mb4\_unicode\_ci**

The screenshot shows the phpMyAdmin interface. The top navigation bar has tabs for Databases, SQL, Status, Users, and Export. The 'Databases' tab is highlighted. Below the tabs, there's a 'General Settings' section with a 'Change password' link and a dropdown for 'Server connection collation' set to 'utf8\_general\_ci'. Underneath is an 'Appearance Settings' section with language and theme options. On the left sidebar, there's a list of databases: information\_schema, mysql, performance\_schema, and testdb.

The screenshot shows the 'Bases de données' (Databases) page in phpMyAdmin. The top menu has tabs for Bases de données, SQL, État, Comptes utilisateurs, Exporter, Importer, and Panneau. The 'Bases de données' tab is active. A sub-section titled 'Création d'une base de données' is shown, with a text input for the database name 'clinic\_management' and a dropdown for encoding 'utf8mb4\_general\_ci'. A 'Créer' (Create) button is visible.

- 4. Click Import in the top menu**

The screenshot shows the 'Importation dans la base de données « clinic\_management »' (Import into database « clinic\_management ») page. The top menu includes Structure, SQL, Rechercher, Requête, Exporter, Importer, Opérations, Priviléges, and Plus. The 'Importeur' (Importer) tab is active. A form allows selecting a file to import. The 'Fichier à importer:' field contains 'clinic\_management.sql'. It also specifies the character set as 'utf-8'.



## Étape 4 : Déploiement des fichiers du projet :

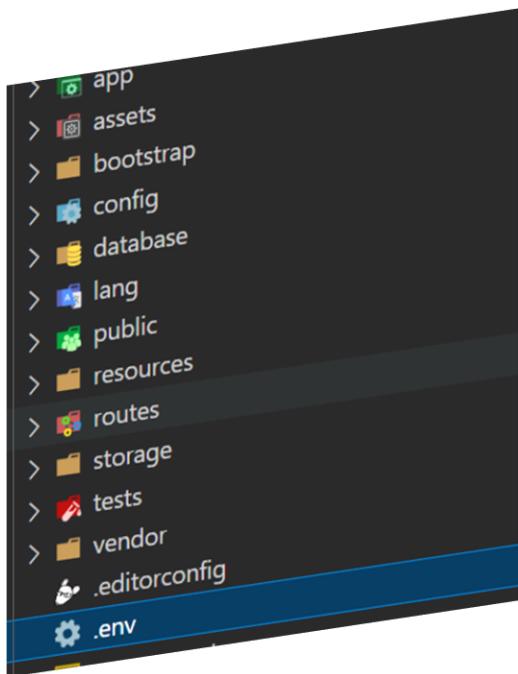
1. Extraire les fichiers du projet dans le dossier créé précédemment
2. Ouvrir un terminal et naviguer vers le dossier du projet
3. Exécuter les commandes suivantes :

```
composer install
npm install
npm run build
cp .env.example .env
php artisan key:generate
```



## Étape 5 : Configuration du fichier .env

Modifier le fichier .env avec les paramètres suivants :



```
APP_NAME="Système de Gestion pour une Clinique Médicale"
APP_ENV=local
APP_DEBUG=true
APP_URL=http://localhost/clinic-management

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=clinic_management
DB_USERNAME=root
DB_PASSWORD=
```



## Étape 6 : Initialisation de la base de données

Exécuter les migrations et les seeders pour créer la structure de la base de données et y insérer les données initiales :

```
php artisan migrate  
php artisan db:seed
```



## Étape 7 : Configuration du serveur web

### 1. Pour un hôte virtuel (recommandé) :

- Créer un fichier de configuration dans le dossier des hôtes virtuels d'Apache
- Configurer le DocumentRoot vers le dossier public du projet
- Activer le module de réécriture d'URL (mod\_rewrite)

### 2. Alternative : Utiliser le serveur de développement intégré à Laravel

```
php artisan serve
```

#### **4.2.3 Déploiement en environnement de production:**

→ Pour déployer le système sur un serveur de production, suivez ces étapes supplémentaires :

→ **Étape 1 : Préparation du serveur:**

- 1. Installer les prérequis sur le serveur (PHP 8.1, MySQL, Nginx/Apache)**
- 2. Configurer un utilisateur MySQL dédié avec des privilèges limités**
- 3. Configurer le pare-feu pour n'autoriser que les ports nécessaires (80, 443)**

→ **Étape 2 : Configuration des services externes:**

**Pour activer les fonctionnalités complètes, configurez les services externes dans le fichier .env :**

- 1. Configuration du service de messagerie :**

```
MAIL_MAILER=smtp
MAIL_HOST=smtp.votreservice.com
MAIL_PORT=587
MAIL_USERNAME=votre_email@exemple.com
MAIL_PASSWORD=votre_mot_de_passe
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS=contact@clinique.com
MAIL_FROM_NAME="${APP_NAME}"
```

## 2. Configuration du service de messagerie :

```
# Stripe  
STRIPE_KEY=pk_test_votre_cle_publique  
STRIPE_SECRET=sk_test_votre_cle_secrete  
  
# PayPal  
PAYPAL_CLIENT_ID=votre_client_id  
PAYPAL_SECRET=votre_secret  
PAYPAL_MODE=sandbox # Changer en "live" pour la production
```

## 3. Configuration du service de messagerie :

```
NOCAPTCHA_SECRET=votre_cle_secrete  
NOCAPTCHA_SITEKEY=votre_cle_site
```

### 4.2.4 Accès au système:

→ Une fois l'installation terminée, le système est accessible avec les identifiants par défaut suivants :

#### 1. Accès administrateur :

URL : [votre-domaine.com/login](http://votre-domaine.com/login) ou [localhost/login](http://localhost/login)

Email : [admin@omarhajjoub.com](mailto:admin@omarhajjoub.com)

Mot de passe : **123456**

#### 2. Accès médecin :

Email : [doctor@omarhajjoub.com](mailto:doctor@omarhajjoub.com)

Mot de passe : **123456**

### 3. Accès patient

**Email :** [patient@omarhajjoub.com](mailto:patient@omarhajjoub.com)

**Mot de passe :** **123456**

### 4.3. Évaluation des objectifs:



Objectif	Évaluation
Automatiser la gestion des rendez-vous	Réussi : Les patients peuvent réserver, annuler et consulter leurs rendez-vous.
Intégrer un système de paiement sécurisé	Réussi : Intégration avec Stripe et PayPal pour les paiements en ligne.
Améliorer l'expérience utilisateur	Réussi : Interfaces simples et responsives grâce à Bootstrap et Blade.
Fournir des statistiques et des rapports	Réussi : Génération de rapports sur les performances et les paiements.
Assurer la sécurité des données	Réussi : Utilisation de la validation des données et des middlewares Laravel.

#### 4.4. Perspectives d'amélioration :



Amélioration	Description
Optimisation des performances	Implémenter un système de caching pour réduire les temps de réponse.
Notifications en temps réel	Intégrer Firebase pour des notifications instantanées et améliorer la réactivité.
Support multilingue	Ajouter la prise en charge de plusieurs langues pour élargir l'audience.
Amélioration de l'interface utilisateur	Refonte de certaines pages avec des frameworks modernes comme Vue.js ou React.
Ajout de nouvelles fonctionnalités	Intégrer un module de gestion des dossiers médicaux pour les patients.
Amélioration de la gestion des rôles	Ajouter des permissions plus granulaires pour les utilisateurs.

## Bibliographie

1. Documentation Laravel : <https://laravel.com/docs>
2. Documentation Blade : <https://laravel.com/docs/blade>
3. Documentation Livewire : <https://laravel-livewire.com/docs>
4. Documentation MySQL : <https://dev.mysql.com/doc/>
5. Bootstrap : <https://getbootstrap.com/>
6. Stripe API : <https://stripe.com/docs/api>
7. PayPal API : <https://developer.paypal.com/docs/api/overview/>
8. PlantUML : <https://plantuml.com/>

## Comparaison avec des systèmes similaires

Afin d'évaluer la performance de notre système de gestion de clinique médicale, nous avons effectué une comparaison avec deux solutions existantes et populaires sur le marché : **Doctolib** et **Zocdoc**.

Cette comparaison met en évidence les forces et les points d'amélioration de notre projet.

Fonctionnalité	Notre système	Doctolib	Zocdoc
⌚ Prise de rendez-vous en ligne	<input checked="" type="checkbox"/> Oui – Simple et rapide	<input checked="" type="checkbox"/> Oui	<input checked="" type="checkbox"/> Oui
💳 Paiement en ligne	<input checked="" type="checkbox"/> Oui – Stripe & PayPal	<input checked="" type="checkbox"/> Non	<input checked="" type="checkbox"/> Carte bancaire uniquement
🌐 Multilingue	<input checked="" type="checkbox"/> Oui (FR / EN / AR)	<input checked="" type="checkbox"/> Oui (FR / EN)	<input checked="" type="checkbox"/> Oui (EN / ES)
📲 Notifications	<input checked="" type="checkbox"/> Email + intégration possible SMS	<input checked="" type="checkbox"/> SMS + Email	<input checked="" type="checkbox"/> SMS
👤 Choix du médecin par spécialité	<input checked="" type="checkbox"/> Oui avec filtres avancés	<input checked="" type="checkbox"/> Oui	<input checked="" type="checkbox"/> Oui
🔒 Sécurité des données	<input checked="" type="checkbox"/> Chiffrement, validation, HTTPS	<input checked="" type="checkbox"/> Conforme RGPD	<input checked="" type="checkbox"/> Conforme HIPAA
📊 Statistiques & rapports	<input checked="" type="checkbox"/> Inclus avec tableaux interactifs	<input checked="" type="checkbox"/> Basique	<input checked="" type="checkbox"/> Très limité
💻 Interface personnalisable pour la clinique	<input checked="" type="checkbox"/> Oui – Open Source et modifiable	<input checked="" type="checkbox"/> Non	<input checked="" type="checkbox"/> Non
💰 Coût d'utilisation	<input checked="" type="checkbox"/> Gratuit / auto-hébergé	<input checked="" type="checkbox"/> Abonnement mensuel élevé	<input checked="" type="checkbox"/> Abonnement mensuel élevé

## Analyse comparative

- **Avantages de notre système :**

- Intégration de **plusieurs passerelles de paiement**.
- Interface **multilingue** adaptée à différents profils d'utilisateurs.
- **Statistiques détaillées** pour les administrateurs.
- Système **entièrement personnalisable** selon les besoins de la clinique.

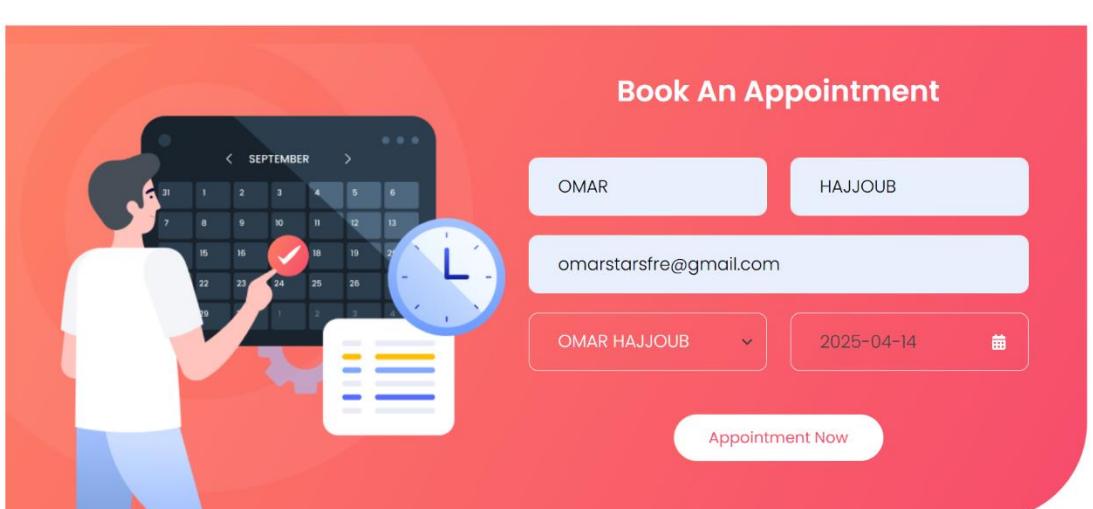
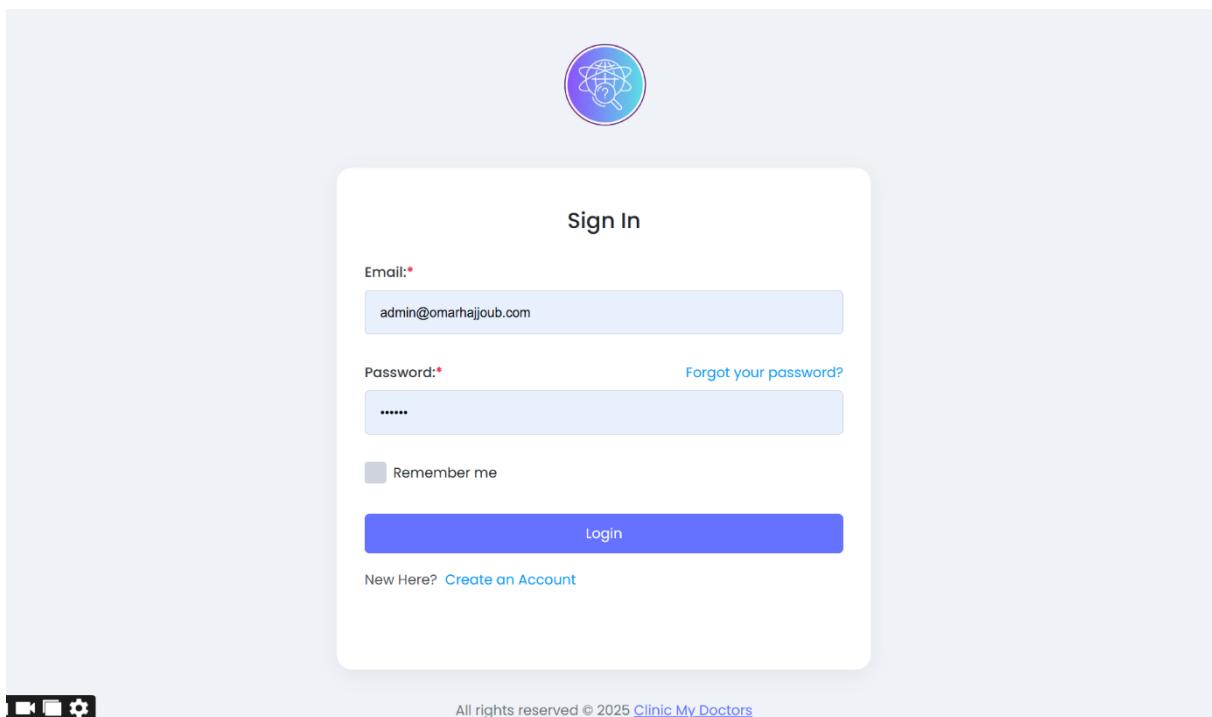
- **Limites actuelles :**

- Absence d'application mobile dédiée (peut être ajoutée ultérieurement).
  - Notifications SMS dépendantes de services externes comme Twilio.
- 

## Conclusion de la comparaison

Cette analyse démontre que notre système propose une **solution complète, flexible et économique**, adaptée aux petites et moyennes cliniques qui souhaitent digitaliser leurs opérations sans dépendre de services externes coûteux.

## Captures d'écran supplémentaires



First Name:<sup>\*</sup>

Last Name:<sup>\*</sup>

Email:<sup>\*</sup>

Doctor:<sup>\*</sup>

Service:<sup>\*</sup>

Contact No:

+212
0655560301

Valid Number

Appointment Date: <sup>\*</sup>

Payment Method: <sup>\*</sup>

Available Slots: Booked Available

08:30 AM – 08:45 AM	08:50 AM – 09:05 AM	09:10 AM – 09:25 AM	09:30 AM – 09:45 AM	09:50 AM – 10:05 AM
10:10 AM – 10:25 AM	10:30 AM – 10:45 AM	10:50 AM – 11:05 AM	11:10 AM – 11:25 AM	11:30 AM – 11:45 AM
11:50 AM – 12:05 PM	12:10 PM – 12:25 PM	12:30 PM – 12:45 PM	12:50 PM – 01:05 PM	01:10 PM – 01:25 PM
01:30 PM – 01:45 PM	01:50 PM – 02:05 PM	02:10 PM – 02:25 PM	02:30 PM – 02:45 PM	02:50 PM – 03:05 PM
03:10 PM – 03:25 PM	03:30 PM – 03:45 PM	03:50 PM – 04:05 PM	04:10 PM – 04:25 PM	04:30 PM – 04:45 PM
04:50 PM – 05:05 PM	05:10 PM – 05:25 PM	05:30 PM – 05:45 PM		

PAYABLE AMOUNT : \$ 500

[Confirm Booking](#)

First Name\*: OMAR

Last Name\*: HAJJOUR

Email\*: omarstarsfre@gmail.com

Doctor: OMAR HAJJOUR

Service: Diagnostics

Contact No: +212 0655560301

Appointment Date: 2025-04-14

Payment Method: Manually

Available Slots: Booked Available

08:30 AM - 08:45 AM	08:50 AM - 09:05 AM	09:10 AM - 09:25 AM	09:30 AM - 09:45 AM	09:50 AM - 10:05 AM
10:10 AM - 10:25 AM	10:30 AM - 10:45 AM	10:50 AM - 11:05 AM	11:10 AM - 11:25 AM	11:30 AM - 11:45 AM
11:50 AM - 12:05 PM	12:10 PM - 12:25 PM	12:30 PM - 12:45 PM	12:50 PM - 01:05 PM	01:10 PM - 01:25 PM
01:30 PM - 01:45 PM	01:50 PM - 02:05 PM	02:10 PM - 02:25 PM	02:30 PM - 02:45 PM	02:50 PM - 03:05 PM
03:10 PM - 03:25 PM	03:30 PM - 03:45 PM	03:50 PM - 04:05 PM	04:10 PM - 04:25 PM	04:30 PM - 04:45 PM
04:50 PM - 05:05 PM	05:10 PM - 05:25 PM	05:30 PM - 05:45 PM		

PAYABLE AMOUNT : \$ 500

Confirm Booking

omarstarsfre@gmail.com

Doctor: \*  
OMAR HAJJOUR

Contact No:  
+212 0655560301

Valid Number

Payment Method: \*  
Manually

Patient Name

Service: \*  
Diagnostics

Appointment Date: \*  
2025-04-14

Available Slots: \* Booked Available

08:30 AM - 08:45 AM	08:50 AM - 09:05 AM	09:10 AM - 09:25 AM	09:30 AM - 09:45 AM	09:50 AM - 10:05 AM
10:10 AM - 10:25 AM	10:30 AM - 10:45 AM	10:50 AM - 11:05 AM	11:10 AM - 11:25 AM	11:30 AM - 11:45 AM
11:50 AM - 12:05 PM	12:10 PM - 12:25 PM	12:30 PM - 12:45 PM	12:50 PM - 01:05 PM	01:10 PM - 01:25 PM
01:30 PM - 01:45 PM	01:50 PM - 02:05 PM	02:10 PM - 02:25 PM	02:30 PM - 02:45 PM	02:50 PM - 03:05 PM
03:10 PM - 03:25 PM	03:30 PM - 03:45 PM	03:50 PM - 04:05 PM	04:10 PM - 04:25 PM	04:30 PM - 04:45 PM
04:50 PM - 05:05 PM	05:10 PM - 05:25 PM	05:30 PM - 05:45 PM		

PAYABLE AMOUNT : \$ 500

Confirm Booking

Dashboard

Dashboard

OMAR HAJJOUR

1 Total Active Doctors
6 Total Patients
0 Today Appointments
0 Today Registered Patients

**Earnings From Appointments (\$ 0)**

Month	Earnings (\$)
Jan	2.0
Feb	1.8
Mar	1.2
Apr	0.8
May	0.4
Jun	0.0
Jul	0.0
Aug	0.0
Sep	0.0
Oct	0.0
Nov	0.0
Dec	0.0

**Recent Patients Registration**

NAME	PATIENT UNIQUE ID	TOTAL APPOINTMENTS	REGISTERED ON
No Data Available			

All Rights Reserved ©2025 Clinic My Doctors v8.0

Type de test	Nombre de tests	Réussis	Échoués	Taux de réussite
Tests unitaires	50	48	2	96%
Tests fonctionnels	20	18	2	90%
Tests d'intégration	10	9	1	90%



## Patient Registration

**First Name:**\*

**Last Name:**\*

**Email:**\*

**Password:**\*

**Confirm Password:**\*

I Agree [Terms and conditions.](#)

**Submit**

Already have an account? [Sign in here](#)

All rights reserved © 2025 Clinic My Doctors



☰ **Staffs**

Q Search

- Dashboard
- Staffs**
- Doctors
- Patients
- Smart Patient Cards
- Appointments
- Medicines
- Transactions
- Visits
- Services
- Specializations
- Enquiries

Add Staff

**First Name:**\* Assistant **Last Name:**\* I

**Email:**\* Assistant1@omarhajjoub.com **Contact No:** +212 - Contact No

**Password:**\* I23456 **Confirm Password:**\* \*\*\*\*\*

**Role:**\* Staff **Gender:**\* Male Female

**Profile:**

**Save** **Discard**

Back  OMAR HAJJOUR


OMAR HAJJOUR

Doctors Doctor Schedules

Search
Back

- [Dashboard](#)
- [Staffs](#)
- [Doctors](#)
- [Patients](#)
- [Smart Patient Cards](#)
- [Appointments](#)
- [Medicines](#)
- [Transactions](#)
- [Visits](#)
- [Services](#)
- [Specializations](#)
- [Enquiries](#)

### Edit Doctor

First Name:\*

Email:\*

DOB:

Experience In Year:

Blood Group:

Last Name:\*

Contact Number:

Valid Number

Specialization:\*

Marine Medicine
Medical Genetics
Microbiology
Nuclear Medicine

Paediatrics
Palliative Medicine
Pathology
Pharmacology

Psychiatry
Physiology
Physical Medicine
Radiotherapy

Select Gender:\*

 Male  Female

### Add Qualification

Degree:\*

University:\*

Year:\*

Save
Discard


Doctors

Doctors   Doctor Schedules

 OMAR HAJJOUR

### Add Doctor

First Name:\*

Last Name:\*

Email:\*

Contact Number:

+212

Password:\*

Confirm Password:\*

DOB:

Specialization:\*

Experience In Year:

Select Gender : \*

 Male  Female

Blood Group:

Twitter:

Back


Doctors

Doctors   Doctor Schedules

 Email verified  OMAR HAJJOUR

DOCTOR	STATUS	EMAIL VERIFIED	IMPERSONATE	REGISTERED ON	ACTION
 doctor I <span>☆☆☆☆</span> doctor@gmail.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<button>Impersonate</button>	13 Apr 2025 09:48 PM	<span>+</span> <span>Edit</span> <span>Delete</span>
 OMAR HAJJOUR <span>☆☆☆☆</span> doctor@omarhajjoub.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<button>Impersonate</button>	13 Apr 2025 09:23 PM	<span>+</span> <span>Edit</span> <span>Delete</span>


Specializations

 OMAR HAJJOUR

NAME	ACTION
Marine Medicine	<span>Edit</span> <span>Delete</span>
Medical Genetics	<span>Edit</span> <span>Delete</span>
Microbiology	<span>Edit</span> <span>Delete</span>
Nuclear Medicine	<span>Edit</span> <span>Delete</span>
Paediatrics	<span>Edit</span> <span>Delete</span>
Palliative Medicine	<span>Edit</span> <span>Delete</span>
Pathology	<span>Edit</span> <span>Delete</span>
Pharmacology	<span>Edit</span> <span>Delete</span>

Sidebar:

- Patients
- Smart Patient Cards
- Appointments
- Medicines
- Transactions
- Visits
- Services
- Specializations
- Enquiries
- Subscribers
- Front CMS**
- Settings

Header: CMS Banner FAQs Front Patient Testimonials

User: OMAR HAJJOUR

Form Fields:

- About Image 1: (File upload)
- About Image 2: (File upload)
- About Image 3: (File upload)
- About Title: \* (Text input: What we do Actually)
- About Experience: \* (Text input: 20)
- About Short Description: \* (Text area: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Consectetur necessitatibus placeat numquam enim adipisci nostrum facilis distinctio modi; cupiditate laborum ea eius repellendus? Obcaecati saepe numquam pariatur aliquid, aspernatur necessitatibus dolores harum quos eum esse, laudantium odit alias, iste dolorem?)
- Terms & Conditions: \* (Text area: Normal)

Sidebar:

- Dashboard
- Staffs
- Doctors
- Patients
- Smart Patient Cards
- Appointments
- Medicines
- Transactions
- Visits
- Services
- Specializations
- Enquiries

Header: CMS Banner FAQs Front Patient Testimonials

User: OMAR HAJJOUR

Table:

IMAGE	TITLE	SHORT DESCRIPTION	ACTION
	We Provide All Health Care Solution	Protect Your Health And Take Care To Of Your Health	

Show 50 Showing 1 results

All Rights Reserved ©2025 Clinic My Doctors

127.0.0.1:8000/admin/banner v8.1.0

Sidebar:

- Dashboard
- Staffs
- Doctors
- Patients
- Smart Patient Cards
- Appointments
- Medicines
- Transactions
- Visits
- Services
- Specializations
- Enquiries

Header: CMS Banner FAQs Front Patient Testimonials

User: OMAR HAJJOUR

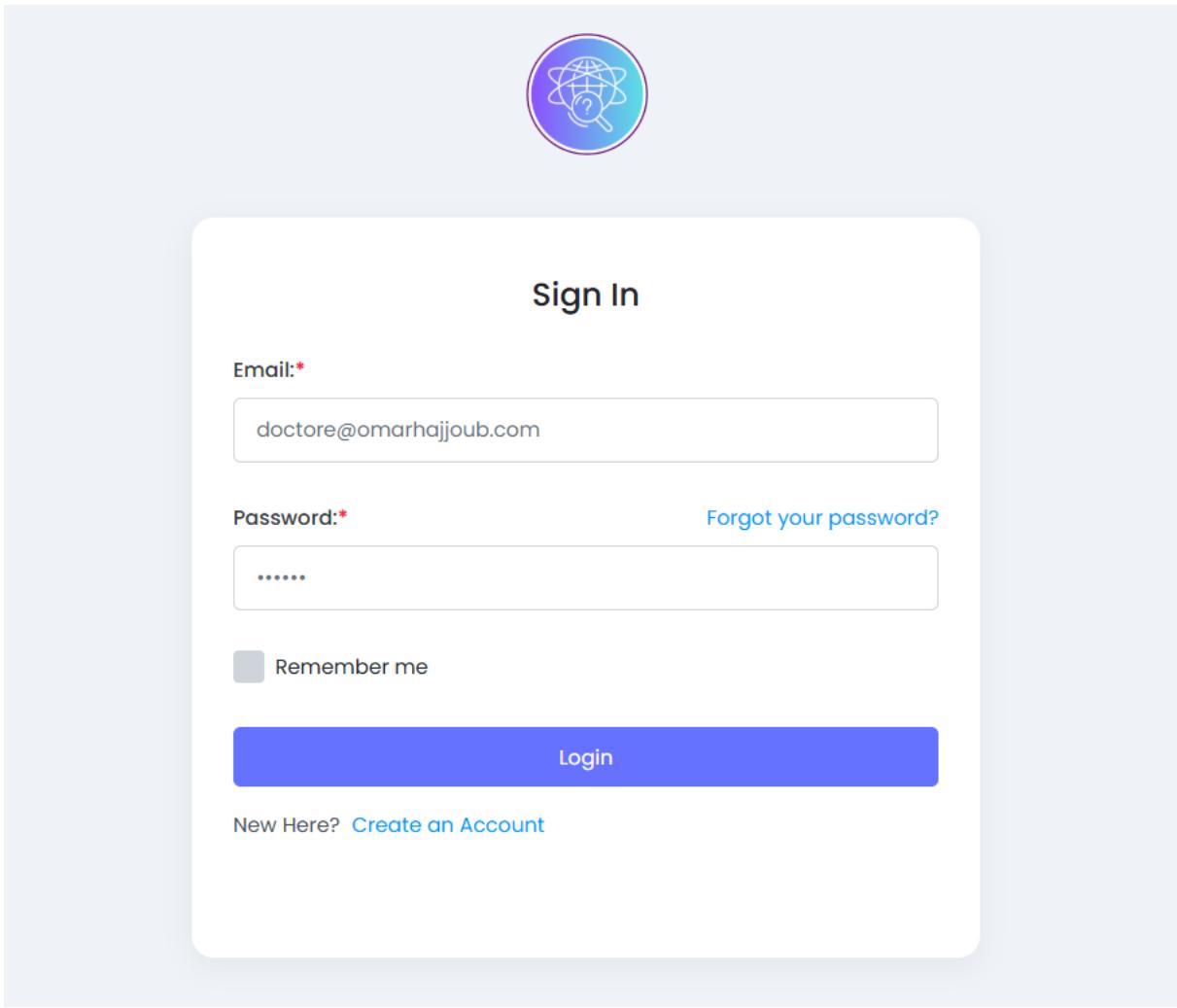
Table:

NAME	SHORT DESCRIPTION	ACTION
JOHN DOE	Incidunt deleniti blanditiis quas aperiam recusandae consequatur ullam quibusdam cum libero illo rerum repellendus!	
COLLIS TATED	Natus voluptatum enim quod necessitatibus quis expedita harum provident eos obcaecati id culpa corporis molestias.	

Show 50 Showing 2 results

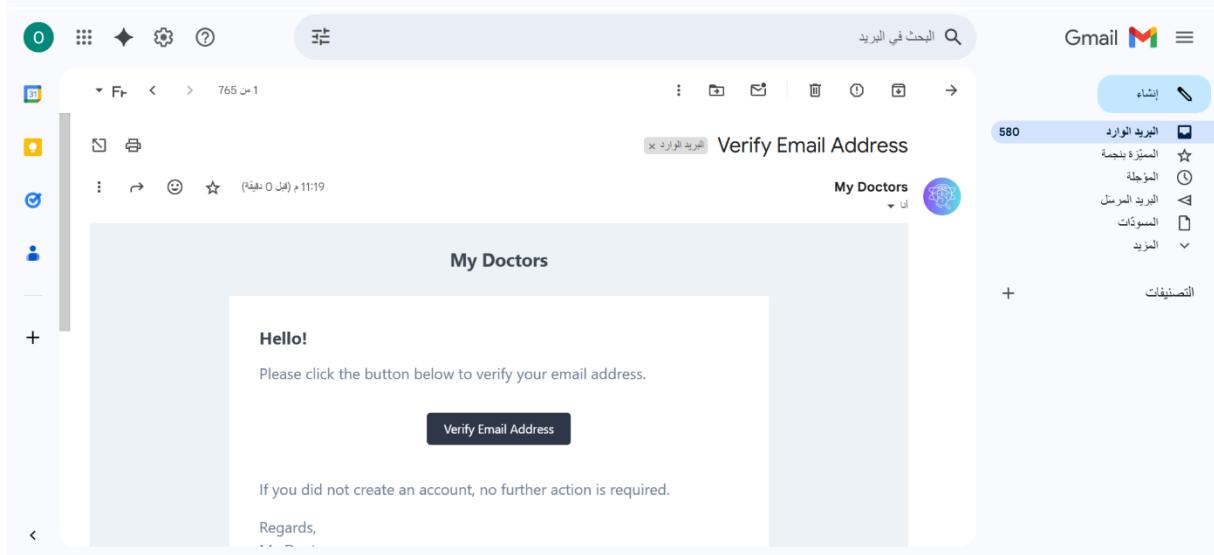
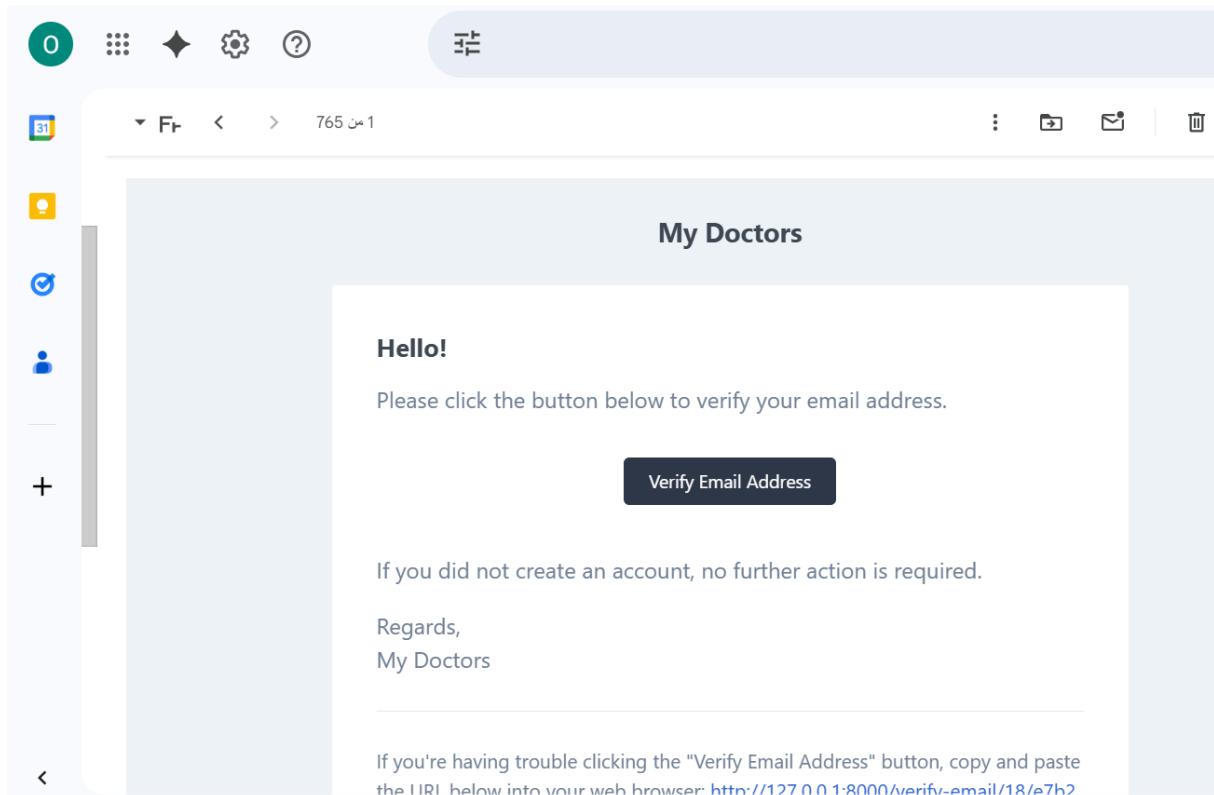
All Rights Reserved ©2025 Clinic My Doctors

v8.1.0



The image shows the 'Dashboard' page of the application. On the left is a sidebar with a search bar at the top and a list of icons: Dashboard (selected), Appointments, Transactions, My Schedule, Visits, Live Consultations, Connect Google Calendar, Holiday, Smart Patient Cards, and Medicines. The main area has three cards: 'Total Appointments' (0), 'Today Appointments' (0), and 'Next Appointments' (0). Below these is a section titled 'Recent Appointments' with a table header: PATIENT, PATIENT UNIQUE ID, and DATE. The table body says 'No Data Available'. At the bottom of the dashboard, there's a footer with 'All Rights Reserved ©2025 Clinic My Doctors' and 'v8.1.0'. The bottom part of the image shows a browser interface with a search bar containing 'البحث في البريد' (Search in email), a message count of 64, and a 'Verify Email Address' button.







## Sign In

Email:<sup>\*</sup>

omarhajjoub85@gmail.com

Password:<sup>\*</sup>

.....

[Forgot your password?](#)

Remember me

[Login](#)

New Here? [Create an Account](#)

Dashboard

Patient 1

Search

Today Appointments: 0

Next Appointments: 0

Completed Appointments: 0

Today Appointments

No Data Available

Upcoming Appointments

DOCTOR DATE

127.0.0.1:8000/patients/appointments


Appointments
  Patient 1

 Dashboard
 Appointments
 Transactions
 Reviews
 Visits
 Live Consultations
 Connect Google Calendar
 Smart Patient Cards

### Add Appointment

Doctor:\*

Date:\*

Available Slots:● Booked ● Available

No time slots found.


Appointments
 Back

 Dashboard
 Appointments
 Transactions
 Reviews
 Visits
 Live Consultations
 Connect Google Calendar
 Smart Patient Cards

### Add Appointment

Doctor:\*

Date:\*

Available Slots:● Booked ● Available

08:30 AM – 08:45 AM	08:50 AM – 09:05 AM	09:10 AM – 09:25 AM	09:30 AM – 09:45 AM	09:50 AM – 10:05 AM
10:10 AM – 10:25 AM	10:30 AM – 10:45 AM	10:50 AM – 11:05 AM	11:10 AM – 11:25 AM	11:30 AM – 11:45 AM
11:50 AM – 12:05 PM	12:10 PM – 12:25 PM	12:30 PM – 12:45 PM	12:50 PM – 01:05 PM	01:10 PM – 01:25 PM
01:30 PM – 01:45 PM	01:50 PM – 02:05 PM	02:10 PM – 02:25 PM	02:30 PM – 02:45 PM	02:50 PM – 03:05 PM
03:10 PM – 03:25 PM	03:30 PM – 03:45 PM	03:50 PM – 04:05 PM	04:10 PM – 04:25 PM	04:30 PM – 04:45 PM
04:50 PM – 05:05 PM	05:10 PM – 05:25 PM	05:30 PM – 05:45 PM		


Appointments
 Back

 Dashboard
 Appointments
 Transactions
 Reviews
 Visits
 Live Consultations
 Connect Google Calendar
 Smart Patient Cards

### Add Appointment

Doctor:\*

Date:\*

Available Slots:● Booked ● Available

08:30 AM – 08:45 AM	08:50 AM – 09:05 AM	09:10 AM – 09:25 AM	09:30 AM – 09:45 AM	09:50 AM – 10:05 AM
10:10 AM – 10:25 AM	10:30 AM – 10:45 AM	10:50 AM – 11:05 AM	11:10 AM – 11:25 AM	11:30 AM – 11:45 AM
11:50 AM – 12:05 PM	12:10 PM – 12:25 PM	12:30 PM – 12:45 PM	12:50 PM – 01:05 PM	01:10 PM – 01:25 PM
01:30 PM – 01:45 PM	01:50 PM – 02:05 PM	02:10 PM – 02:25 PM	02:30 PM – 02:45 PM	02:50 PM – 03:05 PM
03:10 PM – 03:25 PM	03:30 PM – 03:45 PM	03:50 PM – 04:05 PM	04:10 PM – 04:25 PM	04:30 PM – 04:45 PM
04:50 PM – 05:05 PM	05:10 PM – 05:25 PM	05:30 PM – 05:45 PM		

**Gmail** M 

Search mail 

construction 

580 incoming mail  Starred  Postponed  Sent mail  Drafts  More 

Categories 

**My Doctors**  Arabic ← English  View original version 

**My doctors' clinic** 

Hello, Patient 1

Your appointment has been successfully booked for **April 14, 2025**  
between **08:30 AM and 08:45 AM**

.Click the button below to cancel your appointment

**Cancel appointment**

**Patients**

Search 

Dashboard 

Staffs 

Doctors 

**Patients**

Search 

Patient 1  omarhajjoub85@gmail.com  Email Verified  Impersonate  Registered On  13 Apr 2025 10:19 PM  

**Patient Details**

Patient 1  omarhajjoub85@gmail.com N/A  0 Today Appointments  1 Upcoming Appointments  0 Completed Appointments  

Overview Appointments

Blood Group N/A	Gender Female
DOB N/A	Address N/A
Registered On 14 minutes ago	Last Updated 11 minutes ago

All Rights Reserved ©2025 Clinic My Doctors

**Dashboard**

Search 

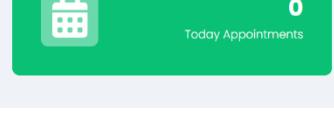
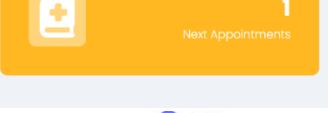
Dashboard 

Appointments 

Transactions 

**Appointments**

Search 

Total Appointments  6  0  1  OMAR HAJJOUR  

**Appointments**

PATIENT  Patient 1  omarhajjoub85@gmail.com  08:30 AM - 08:45 AM  \$500.00  Pending  Booked  

**Appointments**

- [Transactions](#)
- [My Schedule](#)
- [Visits](#)
- [Live Consultations](#)
- [Connect Google Calendar](#)
- [Holiday](#)
- [Smart Patient Cards](#)
- [Medicines](#)

**Appointments**

[Overview](#) [Prescriptions](#)

Appointment ID:	CINBESOUR	Appointment At:	14 Apr 2025 08:30 AM - 08:45 AM
Status:	<span style="color: blue;">Booked</span>	Patient:	Patient 1
Doctor:	OMAR HAJJOUR		
Service:	Diagnostics		
Amount:	\$500.00		
Payment:	Pending		
Created At:	2 minutes ago		

**Calendar**

April 2025

SUN	MON	TUE	WED	THU	FRI	SAT
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3

● Booked ● Check in ● Check out ● Cancelled