

RAPPORT PROJET

Développement d’un Système de Réservation de Terrains en PHP Symfony

Réalisé par :

Badre elmouden

3IIR G22

Encadré par :

Pr.Tace

**Table des matières**

1. **Introduction**-------------------------------------------------3-4  
   1.1 Présentation du Projet-----------------------------------3  
   1.2 Contexte et Problématique------------------------------3  
   1.3 Objectifs---------------------------------------------------3-4
2. **Analyse des Besoins**-----------------------------------------5-6  
   2.1 Cahier des Charges---------------------------------------5  
   2.2 Étude des Utilisateurs------------------------------------5-6  
   2.3 Analyse des Contraintes---------------------------------5-6
3. **Conception du Système**-------------------------------------7-10  
   3.1 Diagramme des Cas d'Utilisation-----------------------7  
   3.2 Diagramme des Classe et de séquence-----------------8-9  
   3.3 Schéma de Base de Données----------------------------10
4. **Développement et Implémentation**-----------------------11-12  
   4.1 Technologies Utilisées-----------------------------------11  
   4.2 Défis Techniques Rencontrés---------------------------11-12
5. **Tests-------------------------------------------**-----------------13-16  
   5.1 Méthodologie de Test parti administrateur-----------13-15
6. 5.2 Méthodologie de Test parti utilisateur----------------15-16
7. **Conclusion et Perspectives**---------------------------------17  
   7.1 Synthèse----------------------------------------------------17  
   7.2 Perspectives d’Amélioration-----------------------------17
8. **Remerciements------------------------------------------------**18

**1. Introduction**:

**1.1 Présentation du Projet**

Le projet consiste à développer un système de réservation de terrains sportifs en ligne à l'aide de PHP et du Framework Symfony. L'objectif est de créer une plateforme accessible via un navigateur web qui permet aux utilisateurs de réserver des terrains de manière rapide et efficace. Le système intégrera une gestion des utilisateurs (clients et administrateurs) et des fonctionnalités pour visualiser, réserver et gérer les créneaux horaires disponibles pour les terrains sportifs.

Le système offrira aussi des outils de gestion pour les administrateurs afin de suivre les réservations, de gérer les utilisateurs et d’assurer une bonne gestion des créneaux horaires. En plus de la gestion des utilisateurs et des réservations, il sera possible de générer des rapports sur l'usage des terrains.

**1.2 Contexte et Problématique**

Actuellement, la réservation de terrains sportifs dans de nombreuses structures se fait de manière traditionnelle, souvent via téléphone ou en personne. Cela entraîne une série de problèmes tels que :

* **Erreurs humaines** : Lors de la prise de réservation, des erreurs de saisie peuvent se produire, telles que des doubles réservations ou des horaires incorrects.
* **Temps perdu** : Les réservations manuelles demandent souvent du temps et de l'attention, ce qui rend le processus inefficace.
* **Accessibilité limitée** : Les utilisateurs doivent attendre des heures d’ouverture pour pouvoir réserver un terrain.
* **Manque de visibilité** : Les structures ont du mal à suivre en temps réel les réservations et la disponibilité des terrains.

Le projet répond à cette problématique en numérisant le processus de réservation, offrant ainsi une solution plus rapide, plus précise et plus accessible, tout en permettant une meilleure gestion pour les administrateurs.

**1.3 objectif:**

L’objectif principal de ce projet est de fournir un système complet de réservation de terrains sportifs qui :

* **Automatise le processus de réservation** : Permettre aux utilisateurs de réserver en ligne, de consulter la disponibilité en temps réel et de recevoir des confirmations automatiques.
* **Permet une gestion efficace des réservations et des utilisateurs** : Les administrateurs peuvent gérer les réservations, voir les créneaux libres ou réservés, et gérer les profils des utilisateurs (inscriptions, annulations, etc.).
* **Offre une interface intuitive pour les clients et les administrateurs** : La plateforme doit être facile à utiliser, que ce soit pour un utilisateur cherchant à réserver un terrain ou pour un administrateur cherchant à gérer les créneaux horaires, les utilisateurs et les réservations.
* **Optimise la gestion des terrains et des horaires** : Réduire les conflits de réservation et maximiser l'utilisation des terrains en optimisant les horaires de disponibilité.
* **Assure un suivi des performances** : Offrir des outils d'analyse permettant aux administrateurs de suivre les tendances de réservation et de prendre des décisions stratégiques basées sur ces données.

### ****2. Analyse des Besoins****

#### **2.1 Cahier des Charges**

Le cahier des charges décrit les fonctionnalités du système et les besoins fonctionnels et techniques pour la création du projet de réservation de terrains.

1. **Fonctionnalités principales pour les utilisateurs** :
   * **Création de compte** : Les utilisateurs doivent pouvoir s'inscrire avec leur email et un mot de passe sécurisé.
   * **Connexion** : Les utilisateurs doivent pouvoir se connecter à leur compte avec leur email et mot de passe.
   * **Réservation de terrains** : Après connexion, les utilisateurs doivent pouvoir réserver un terrain disponible pour un créneau horaire.
   * **Consultation et gestion des réservations** : Les utilisateurs peuvent consulter, modifier ou annuler leurs réservations.
2. **Fonctionnalités principales pour les administrateurs** :
   * **Gestion des utilisateurs** : Les administrateurs doivent être capables de consulter, ajouter, suspendre, réactiver ou supprimer des utilisateurs.
   * **Gestion des réservations** : L’administrateur peut consulter les réservations, les valider, les modifier ou les annuler.
   * **Gestion des créneaux et terrains** : L’administrateur peut gérer la disponibilité des terrains et les horaires disponibles pour les utilisateurs.
   * **Base de données robuste** : Une base de données relationnelle doit être conçue pour stocker les informations des utilisateurs, des terrains, des réservations et des créneaux horaires.

#### **2.2 Étude des Utilisateurs**

Le projet comporte deux types d’utilisateurs avec des besoins et des permissions différentes.

1. **Client (Utilisateur final)** :
   * **Inscription et Connexion** : Le client peut créer un compte avec une adresse email et un mot de passe sécurisé, puis se connecter pour accéder à son tableau de bord.
   * **Réservation** : Une fois connecté, le client peut consulter la disponibilité des terrains et réserver un créneau horaire.
   * **Gestion des réservations** : Le client peut consulter l’historique de ses réservations et annuler ou modifier les réservations existantes.
   * **Gestion du profil** : Le client peut modifier ses informations personnelles, telles que son email et son mot de passe.
2. **Administrateur** :
   * **Gestion des utilisateurs** : L’administrateur peut voir tous les utilisateurs inscrits, les activer/désactiver, les supprimer ou modifier leurs informations (par exemple, email ou mot de passe).
   * **Gestion des réservations** : L’administrateur peut consulter toutes les réservations effectuées, les valider, les modifier ou les annuler. L’administrateur peut également gérer la disponibilité des terrains (par exemple, ajouter ou supprimer des créneaux horaires).
   * **Gestion des terrains** : L’administrateur peut ajouter, modifier ou supprimer des terrains disponibles à la réservation.
   * **Statistiques et rapports** : L’administrateur peut générer des rapports sur l’utilisation des terrains, les réservations passées et les tendances de réservation.

2.3 Analyse des Contraintes

Lors du développement du projet, plusieurs contraintes doivent être prises en compte :

**Utilisation d’outils open-source** : Le projet doit s’appuyer uniquement sur des outils open-source, ce qui inclut le choix du Framework (Symfony), des bibliothèques PHP, du SGBD (Système de Gestion de Base de Données), etc. Cette contrainte garantit la liberté d’utilisation et la réduction des coûts liés aux licences logicielles.

**Sécurité de l’authentification** : L’un des aspects les plus importants du projet est de garantir la sécurité des informations des utilisateurs. Cela inclut l’utilisation de protocoles sécurisés (HTTPS), et la gestion des sessions de manière sécurisée pour éviter les attaques de type session fixation ou vol de session.

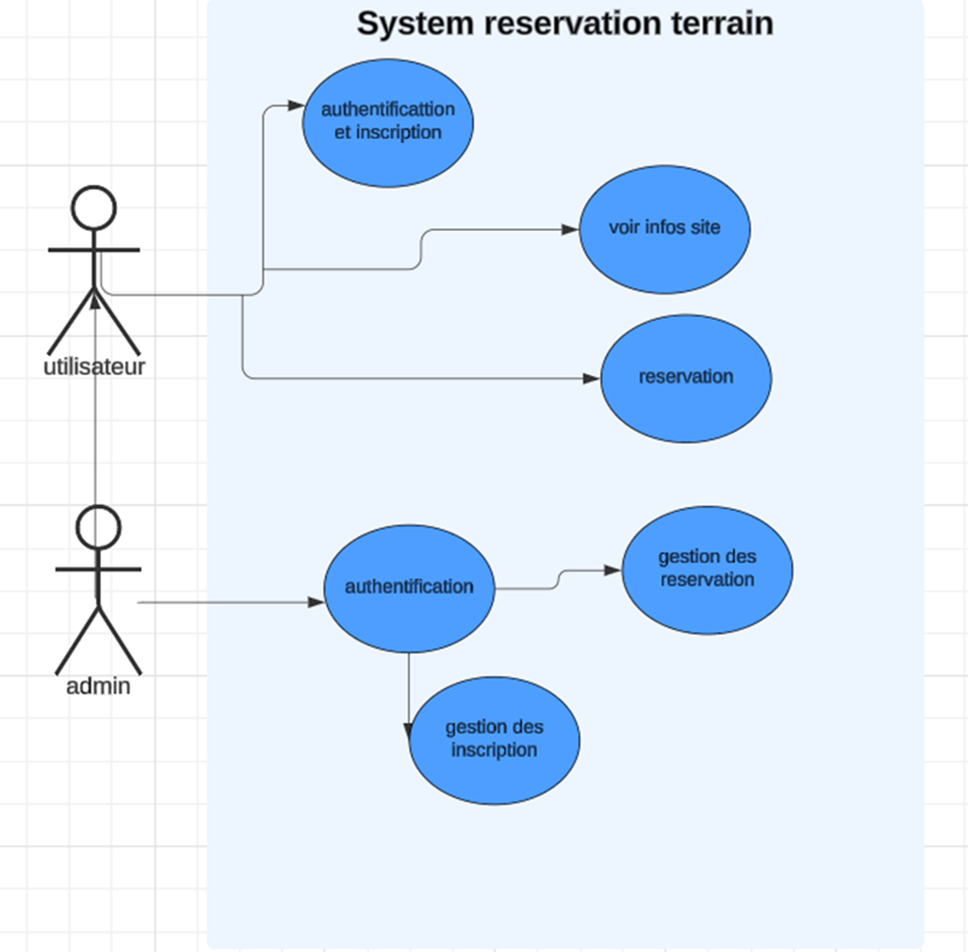
**Disponibilité et performance** : Le système doit être performant, notamment lors de la gestion de réservations en temps réel. L’application devra être conçue pour supporter un grand nombre de requêtes simultanées, notamment lors des périodes de forte demande.

**Scalabilité** : Le système doit être scalable, c’est-à-dire qu’il doit pouvoir supporter une augmentation du nombre d’utilisateurs et de réservations sans compromettre ses performances. Cela inclut une gestion optimale de la base de données et des serveurs.

**Accessibilité et compatibilité multiplateforme** : Le système doit être accessible sur tous les types de dispositifs (ordinateurs de bureau, tablettes, smartphones), avec une interface qui s'adapte bien aux différentes tailles d'écran et navigateurs.

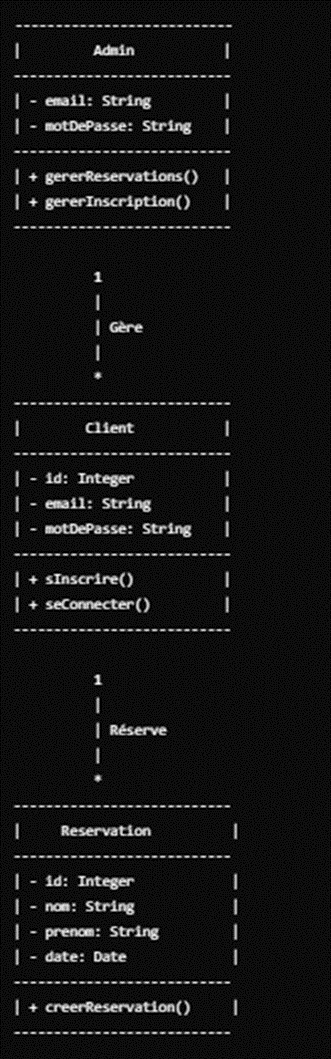
## **3. Conception du Système**

### ****3.1 Diagramme des Cas d'Utilisation****

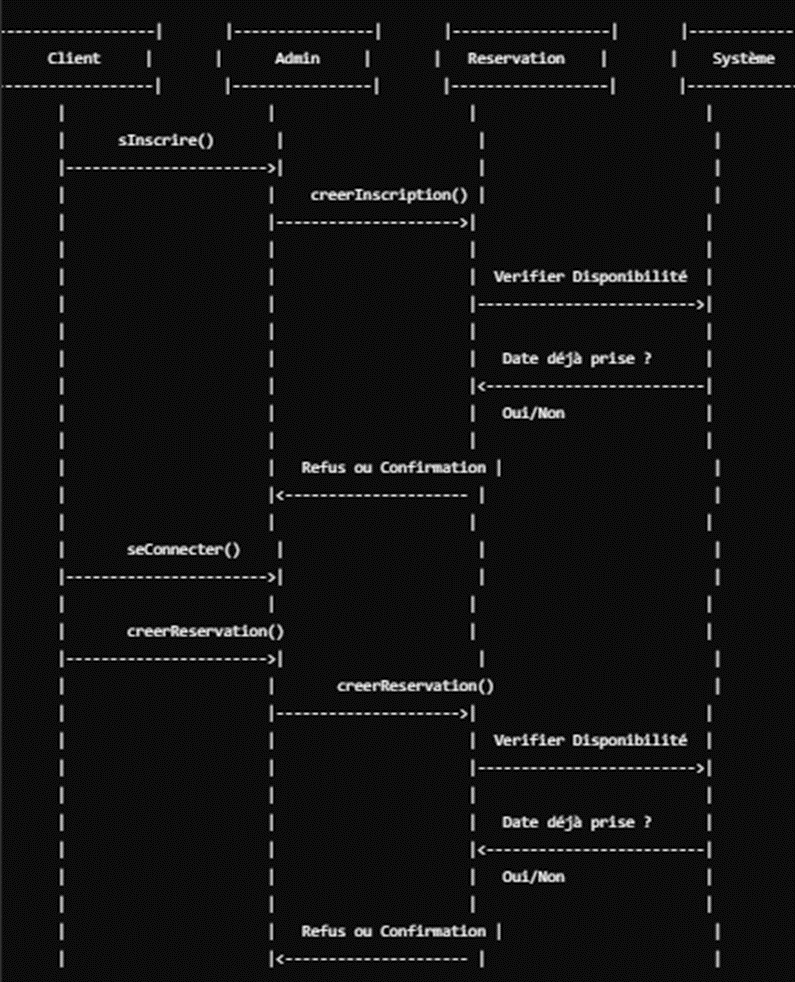
Le diagramme des cas d’utilisation décrit les interactions entre les utilisateurs (clients et administrateurs) et le système. Il inclut les actions principales telles que l’inscription, la réservation et la gestion des réservations.

**3.2 Diagramme des Classe** et de séquence

 **Diagramme de classe** :  
Un diagramme de classe est un type de diagramme utilisé en UML (Unified Modeling Language) pour représenter les différentes classes d'un système ainsi que leurs attributs, méthodes et les relations entre elles. Il décrit la structure statique du système en modélisant les objets, leurs propriétés et les interactions de manière abstraite.

Pour ce projet:

 **Diagramme de séquence** :  
Un diagramme de séquence est un diagramme UML qui montre comment les objets interagissent entre eux dans un scénario particulier. Il illustre l’ordre des messages échangés entre les objets, en représentant l'ordre temporel des événements et les appels de méthodes dans une séquence spécifique d'exécution du système. Ce diagramme est utile pour visualiser la dynamique des interactions dans un processus.

Pour ce projet:

### ****3.3 Schéma de Base de Données****:

Une **base de données** est un ensemble organisé de données stockées et gérées de manière à être facilement accessibles, modifiées et mises à jour. Elle permet de stocker des informations structurées de manière à ce qu'elles puissent être facilement consultées

Voici les entité utilisé dans ce projet:



## **4. Développement et Implémentation**

### ****4.1 Technologies Utilisées****

Le développement du système de réservation de terrains s’appuie sur un ensemble de technologies modernes qui garantissent robustesse, évolutivité et maintenabilité :

* **Langage de programmation :** PHP  
  Utilisé pour le développement côté serveur grâce à sa popularité, sa documentation riche et sa compatibilité avec Symfony.
* **Framework :** Symfony  
  Un framework PHP robuste qui facilite la création d’applications web complexes en proposant une architecture MVC, des outils intégrés comme Twig pour les templates, et un gestionnaire de dépendances.
* **Base de données :** MySQL  
  Utilisé pour stocker les données des utilisateurs, des administrateurs et des réservations de manière relationnelle.
* **Interface utilisateur :** HTML, CSS, et JavaScript  
  Fournissent une interface web simple et intuitive permettant aux utilisateurs d'interagir avec le système.
* **Environnement de développement :**
  + Serveur local : XAMPP pour tester l’application localement.

### ****4.2 Défis Techniques Rencontrés****

Le développement a fait face à plusieurs défis techniques qui ont nécessité des solutions adaptées:

#### **Conflits de réservation**

* **Problème :** Empêcher deux utilisateurs de réserver le même terrain pour le même créneau horaire.
* **Solution :** Ajout d’une validation dans le backend pour vérifier la disponibilité du terrain avant la confirmation de la réservation.

#### **Gestion des rôles utilisateurs (client et administrateur)**

* **Problème :** Assurer une séparation claire entre les permissions des administrateurs et des clients.
* **Solution :** Mise en place du système de rôle intégré dans Symfony, avec des contrôles stricts dans les contrôleurs et la base de données.

#### **Optimisation des requêtes à la base de données**

* **Problème :** La gestion de grandes quantités de données pouvait ralentir le système, en particulier pour les recherches et l'affichage des réservations.
* **Solution :** Utilisation des QueryBuilders de Doctrine pour construire des requêtes optimisées et éviter les jointures inutiles.

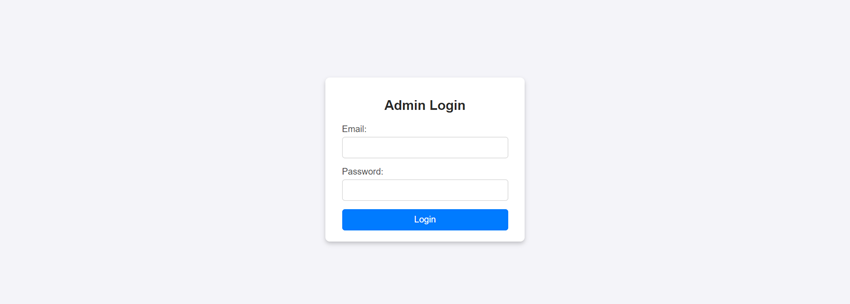
#### **Design réactif pour l’interface utilisateur**

* **Problème :** Garantir une expérience utilisateur fluide sur différentes tailles d’écran (PC, tablette, mobile).
* **Solution :** Utilisation de frameworks CSS comme Bootstrap et de media queries pour adapter l’affichage selon les appareils.

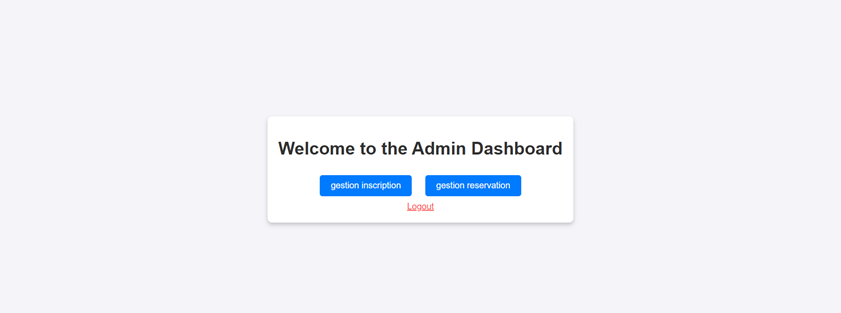
**6.Tests et Validation**

6.1 Méthodologie de Test parti administrateur

1-Admin login: email et mot de passe unique

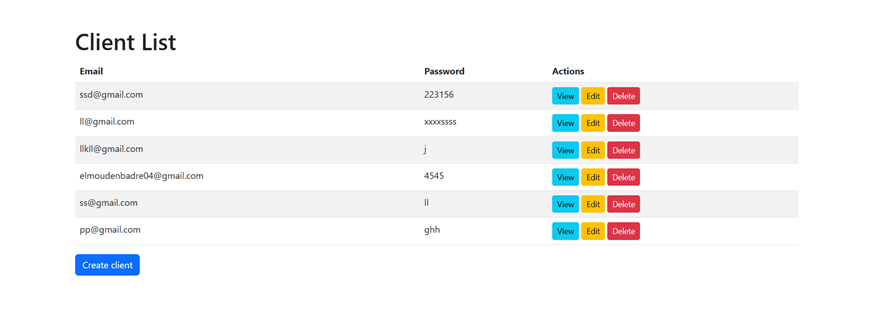


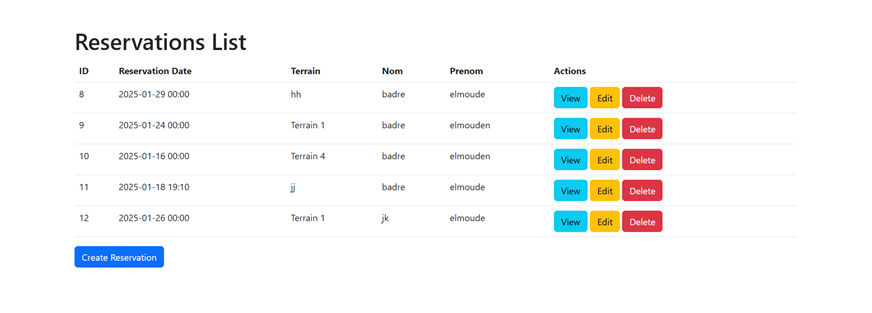
2-Choix de gestion:

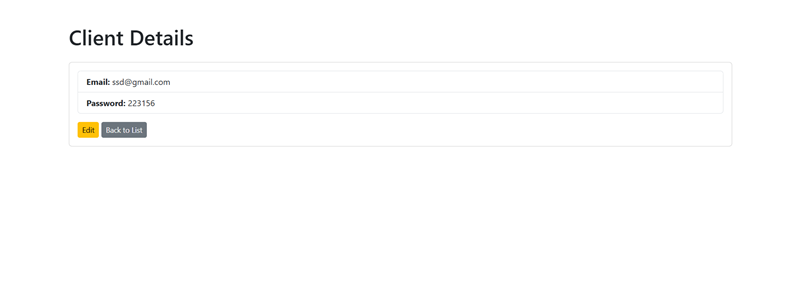


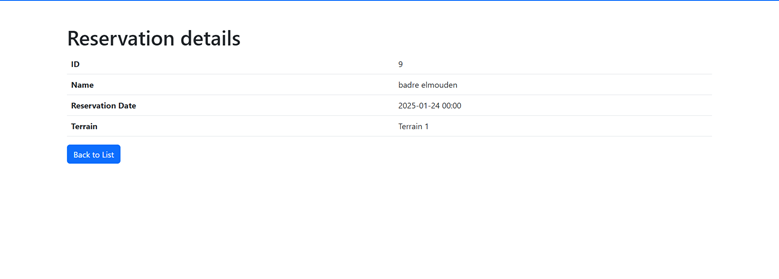
3-système CRUDE:

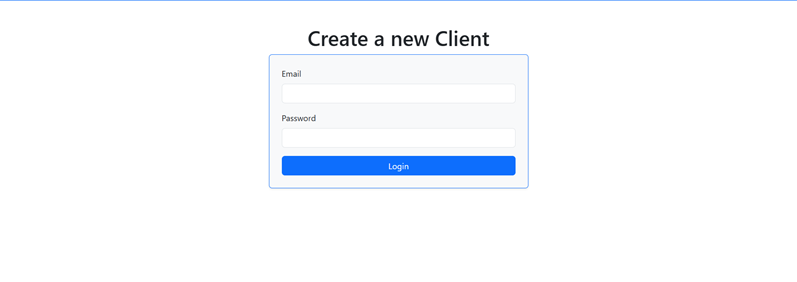
Le système **CRUD** (Create, Read, Update, Delete) représente les quatre opérations de base nécessaires pour gérer les données dans un système ou une base de données. Ces fonctionnalités sont fondamentales dans le développement d’applications. La création (**Create**) permet d’ajouter de nouvelles données, comme l’enregistrement d’un utilisateur ou d’une réservation. La lecture (**Read**) consiste à récupérer et afficher ces données, comme consulter les terrains disponibles ou les réservations. La mise à jour (**Update**) permet de modifier des informations existantes, par exemple changer les détails d’une réservation. Enfin, la suppression (**Delete**) permet de retirer des données du système, comme annuler une réservation ou supprimer un compte. Ces opérations sont généralement implémentées à l’aide de requêtesSQL telles que INSERT, SELECT, UPDATE et DELETE, et constituent la base de nombreux systèmes de gestion modernes

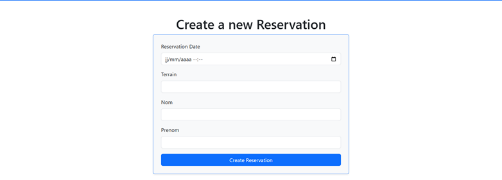


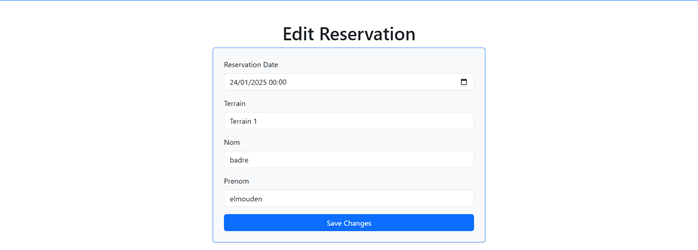


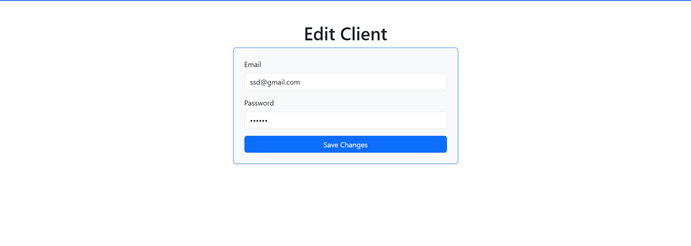


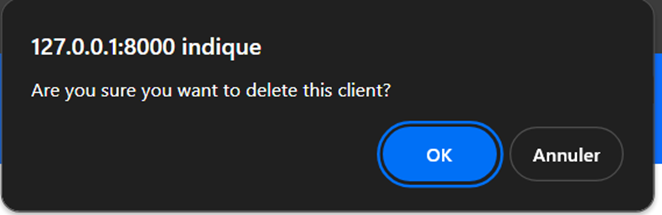


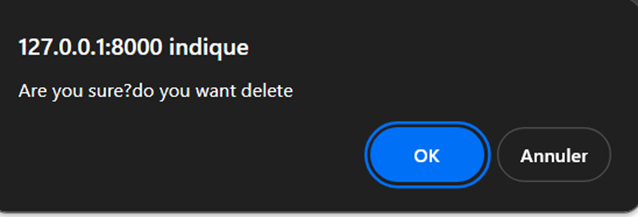






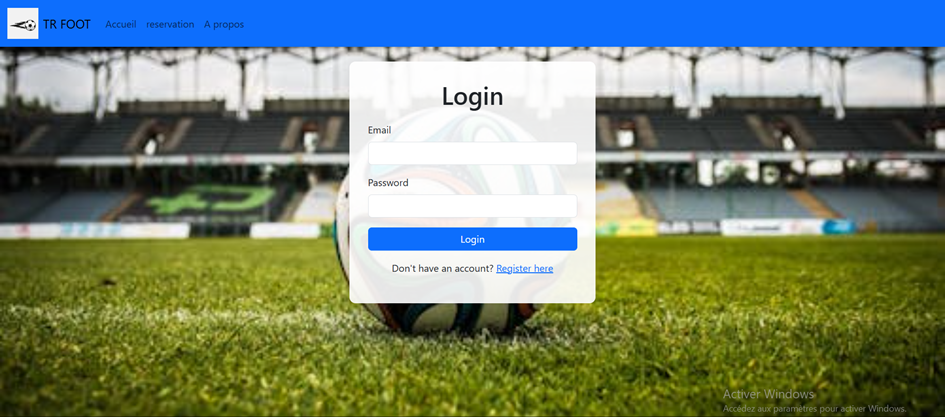


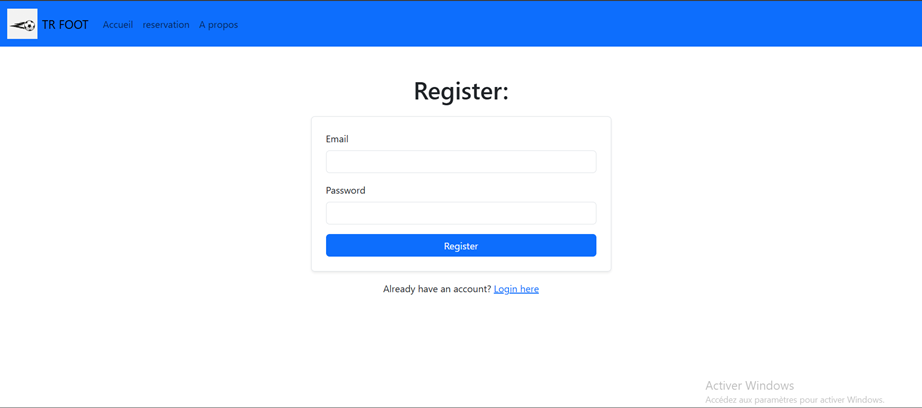


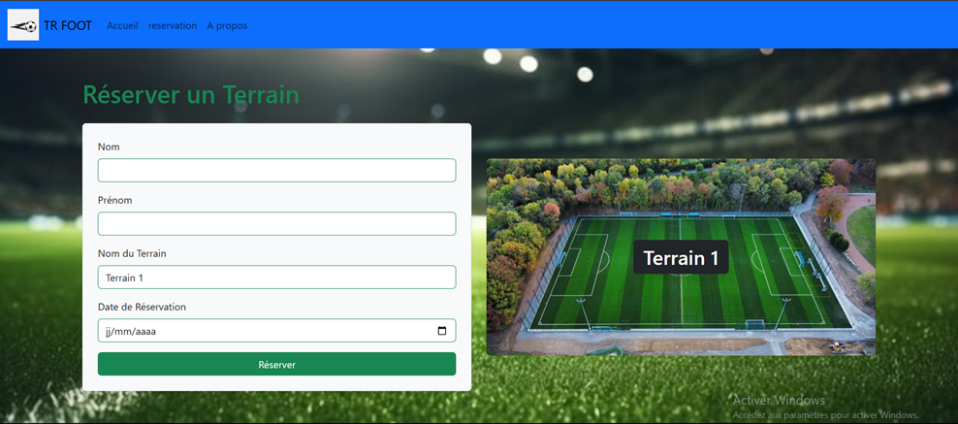


6.1 Méthodologie de Test parti utilisateur

Login utilisateur : Si l'email est incorrect, l'utilisateur est redirigé vers la page d'inscription où il peut créer un nouveau compte. Si le mot de passe est incorrect, un message d'erreur en rouge s'affiche. Dans le cas contraire, l'utilisateur est redirigé vers une autre page



Inscription utilisateur : Si l'email est déjà trouvé dans la base de données, un message d'erreur s'affiche pour informer que l'adresse email est déjà utilisée

Page réservation : Une photo change automatiquement à intervalles réguliers grâce à une fonction JavaScript. Si une date est déjà enregistrée dans le système, un message s'affiche demandant à l'utilisateur de saisir une autre date

Page a propos: contient des informations du site



### ****7. Conclusion et Perspectives****

#### **7.1 Synthèse**

Le projet de développement d'un système de réservation de terrains en PHP Symfony a permis d'automatiser et de simplifier le processus de réservation pour les utilisateurs et les administrateurs. Grâce à l'implémentation des fonctionnalités essentielles, telles que la gestion des utilisateurs, la réservation de terrains et l'administration des données, le système répond aux objectifs initiaux. L'utilisation de Symfony a facilité le développement grâce à son architecture MVC et à ses outils performants, garantissant ainsi une application robuste, sécurisée et évolutive. Les tests réalisés ont confirmé la fiabilité du système dans divers scénarios d'utilisation. Malgré quelques défis techniques rencontrés, comme la configuration de l’authentification JWT ou la gestion des conflits de réservation, ces obstacles ont été surmontés avec succès.

#### **7.2 Perspectives d’Amélioration**

Bien que le projet soit fonctionnel, plusieurs améliorations peuvent être envisagées :

1. **Amélioration de l'interface utilisateur** : Rendre l'interface plus conviviale et moderne grâce à des frameworks front-end comme React ou Vue.js.
2. **Ajout de notifications** : Mettre en place un système de notification (email ou SMS) pour informer les utilisateurs des confirmations ou modifications de leurs réservations.
3. **Gestion des paiements** : Intégrer une plateforme de paiement en ligne pour permettre aux utilisateurs de payer directement leurs réservations.
4. **Rapports avancés** : Ajouter une fonctionnalité permettant aux administrateurs de générer des rapports détaillés sur l'utilisation des terrains et les réservations.
5. **Accessibilité mobile** : Optimiser l'application pour une utilisation sur mobile ou développer une version native pour les appareils mobiles.
6. **Multi-langues** : Permettre le support de plusieurs langues pour rendre le système accessible à un public plus large.

**8-**Remerciments

Je tiens à exprimer ma sincère gratitude à toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce projet.

Tout d’abord, je remercie chaleureusement mon encadrant, Monsieur tace, pour son soutien constant, ses conseils précieux, et sa disponibilité tout au long du projet.

Je remercie également mes camarades de classe pour leurs retours constructifs et leur encouragement.

Enfin, je remercie l’EMSI pour m’avoir offert l’opportunité de travailler sur ce projet enrichissant