

LST : Ingénierie de Développement d'Applications Informatiques

Module : Développement Web

Département : Génie Informatique

RAPPORT DE MINI-PROJET :

Application Web de Réservation des Ressources d'un Data Center

Réalisé par :

- EL BOURMAKI Salim
- EL HAJIOUI Houssam
- DANY Homam
- FARSSI Fatima Zahra

Sous l'encadrement de :

Pr. AIT KBIR Mhamed
Pr. ELYUSUFI Yasyn

Année universitaire : 2025-2026

1. Introduction

2. Analyse et Conception

2.1. Identification des acteurs et Diagramme Use Case

2.2. Modélisation de la base de données

3. Choix Technologiques

3.1. Environnement de Développement:

3.2. Backend : Framework Laravel et PHP.

3.3. Persistance des données : MySQL et Eloquent ORM.

3.4. Frontend : HTML5, CSS et JS natifs (sans frameworks externes).

4. Architecture et Fonctionnalités Implémentées

4.1. Gestion de l'authentification et des accès (Middleware).

4.2. Système de réservation et gestion des conflits (Overlapping).

4.3. Gestion du parc matériel (Serveurs, VM, Stockage).

4.4. Système de notifications et suivi des demandes.

5. Réalisation et Démonstration (Captures d'écran)

5.1. Interfaces utilisateurs et tableaux de bord.

5.2. Gestion administrative et statistiques.

6. Conclusion

1. Introduction

L'infrastructure d'un Data Center représente un ensemble complexe de ressources matérielles et logicielles qui nécessitent une gestion rigoureuse. Ce projet consiste à concevoir et développer une application web robuste avec le framework **Laravel**, dédiée à la réservation et au suivi des ressources informatiques telles que les serveurs physiques, les machines virtuelles et les équipements réseau.

L'objectif principal est d'offrir une plateforme centralisée permettant :

- **Aux utilisateurs** de soumettre des demandes de réservation justifiées et d'en suivre l'évolution en temps réel.
- **Aux responsables techniques** de superviser leur parc matériel et de valider les accès.
- **À l'administrateur** de piloter l'ensemble du système via des statistiques et une gestion granulaire des droits.

Dans un environnement où l'optimisation des ressources est cruciale, cette solution vise à automatiser la vérification des disponibilités et à assurer une traçabilité complète de chaque action effectuée sur le parc.



2. Analyse et Conception

Cette phase définit les fondations du projet en identifiant qui utilise le système, comment ils interagissent avec lui, et comment les données sont structurées.

2.1. Identification des acteurs et diagramme de Use Case :

L'application distingue quatre profils, chacun ayant des responsabilités spécifiques pour garantir une gestion fluide du Data Center :

➤ Invité :

- ✓ Consulte les ressources disponibles en lecture seule.
- ✓ Prend connaissance des règles d'utilisation.
- ✓ Peut déposer une demande de création de compte.

➤ Utilisateur interne (Ingénieur / Enseignant / Doctorant) :

- ✓ Visualise les ressources via des filtres.
- ✓ Effectue des demandes de réservation avec période et justification.
- ✓ Suit l'état de ses demandes (En attente, Active, Terminée, etc.).
- ✓ Signale les incidents techniques.

➤ Responsable technique :

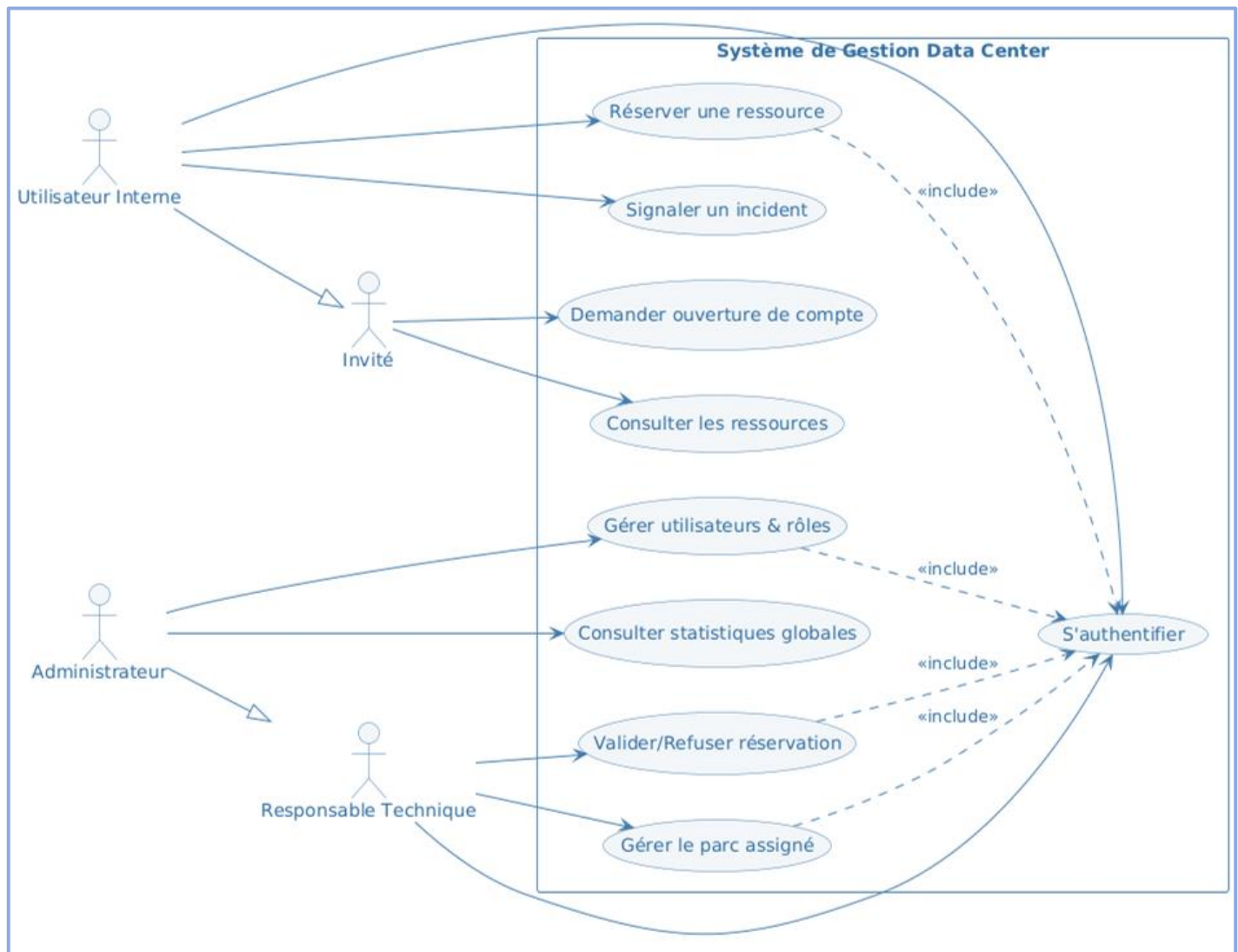
- ✓ Gère le cycle de vie des ressources supervisées (ajout, modification, mise en maintenance).
- ✓ Approuve, planifie ou refuse les réservations avec justification.
- ✓ Modère les discussions et alertes liées à ses ressources.

➤ Administrateur du Data Center :

- ✓ Assure la gestion complète des utilisateurs, des rôles et des permissions.
- ✓ Gère le catalogue global et les périodes de maintenance planifiée.
- ✓ Supervise les statistiques et le taux d'occupation global.

➤ Diagramme Use Case :

Le diagramme suivant synthétise les interactions entre les acteurs identifiés et les fonctionnalités majeures du système de gestion.



2.2. Modélisation de la base de données

La conception de la base de données repose sur une architecture relationnelle permettant une gestion fluide des réservations et des droits d'accès.

➤ Le Modèle Conceptuel des Données (MCD) :

Le MCD permet de visualiser les règles de gestion du système :

- ✓ **Un Utilisateur** peut effectuer plusieurs **Réservations**.

- ✓ **Une Ressource** peut faire l'objet de plusieurs **Réservations** (à des dates différentes).
- ✓ **Un Responsable technique** supervise un ensemble de **Ressources**.
- ✓ **Un Utilisateur** peut signaler plusieurs **Incidents** sur une ressource spécifique.

➤ **Le Modèle Logique des Données (MLD) :**

Le MLD (ou schéma relationnel) traduit ces relations en tables techniques avec leurs clés primaires et étrangères.

- **Analyse des tables clés d'après la réalisation :**

- ✓ **Table *users*** : Gère les comptes et les rôles (Invité, Interne, Responsable, Admin).
- ✓ **Table *resources*** : Contient les spécifications techniques (CPU, RAM, stockage, OS) et l'ID du responsable (*manager_id*).
- ✓ **Table *reservations*** : Table pivot incluant les dates de début/fin, le motif (justification) et le statut de validation.
- ✓ **Table *incidents*** : Permet la traçabilité des problèmes techniques signalés par les utilisateurs.
- ✓ **Table *logs*** : Assure la journalisation de toutes les actions importantes pour l'administrateur.

➤ **Le Modèle Physique des Données (MPD) :**

Voilà la visualisation de la base de données *datacenter_db* dans la figure suivante :

En globale, on a plusieurs tables :

- *Users*, *reservations*, *logs*, *resources* et *incidents* sont liés à travers les clés primaires et étrangères.
- *Failed_jobs*, *notifications*, *migration*, *jobs*, *password_resets*, *personal_acces_tokens*



3. Choix Technologiques :

Cette section justifie l'utilisation des technologies imposées et conseillées pour répondre aux besoins de gestion du Data Center.

3.1. Environnement de Développement:

Pour la mise en place et le test de l'application, nous avons utilisé un environnement local standard :



- **XAMPP** : Cette pile logicielle a servi de serveur local pour exécuter l'application Laravel via **Apache** et gérer le serveur de base de données **MySQL**.



- **phpMyAdmin** : Cet outil d'administration web a été indispensable pour visualiser la structure de la base de données en temps réel, tester les requêtes SQL et vérifier l'insertion correcte des données de réservation.



- **Visual Studio Code** : L'éditeur de texte principal utilisé pour le codage du PHP, du CSS personnalisé et du JavaScript natif.



- **GitHub** : Plateforme de gestion de versions (VCS) basée sur Git. Utilisée pour le suivi des modifications du code source, la gestion des sauvegardes et l'historique du développement, garantissant ainsi la sécurité et la traçabilité du projet.



3.2. Backend : Framework Laravel et PHP.

Le choix de **Laravel** comme framework principal repose sur sa capacité à gérer des applications professionnelles de manière structurée :

- **Architecture MVC** : Permet une séparation nette entre la gestion des ressources (Modèles), la logique de réservation (Contrôleurs) et l'affichage (Vues).
- **Système de Routage** : Centralisation de toutes les URL de l'application dans le fichier web.php pour une maintenance simplifiée.

- **Sécurité Native** : Utilisation des Middlewares pour protéger les accès selon les rôles et sécuriser les formulaires contre les attaques CSRF ou les accès non autorisés.
- **Validation des formulaires** : Utilisation du moteur de validation natif pour garantir la conformité des données de réservation (dates, justificatifs) avant traitement.

3.3. Persistance des données : MySQL et Eloquent ORM.



Pour assurer le stockage et la cohérence des données techniques du parc informatique, nous avons utilisé un système de gestion de base de données relationnelle couplé aux outils de **Laravel** :

- **MySQL** : Choix robuste pour gérer les relations entre les utilisateurs, les équipements et les réservations.
- **Eloquent ORM** : Utilisé pour manipuler la base de données via des modèles PHP, facilitant la gestion des relations complexes, comme le fait qu'un utilisateur possède plusieurs réservations.
- **Migrations** : Mise en place du schéma de données de manière versionnée, permettant de définir les caractéristiques techniques des ressources (CPU, RAM, OS, etc.) directement en code.

3.4. Frontend : HTML5, CSS et JavaScript natifs (sans frameworks externes) :



Conformément aux contraintes strictes du projet, l'interface a été développée sans aucune bibliothèque externe comme Bootstrap, Tailwind ou jQuery :

- **CSS Personnalisé** : Création d'un design responsive et ergonomique pour permettre aux utilisateurs d'effectuer des réservations sur différents supports.
- **JavaScript Natif** : Implémentation de l'interactivité pour le suivi des demandes en temps réel et la gestion des notifications.
- **Interface simple** : Conception d'une expérience utilisateur fluide pour la consultation du catalogue et la soumission des formulaires.

4. Architecture et Fonctionnalités Implémentées

Pour cette partie on va détailler la mise en œuvre technique des fonctionnalités clés permettant d'assurer une gestion efficace et transparente des ressources du Data Center.

4.1. Gestion de l'authentification et des accès (Middleware).

La sécurité de l'application repose sur le système d'authentification natif de **Laravel**, renforcé par une gestion granulaire des droits :

- **Contrôle d'accès** : Utilisation de **Middlewares** personnalisés pour restreindre l'accès aux routes selon le profil de l'utilisateur (Invité, Interne, Responsable, Administrateur).
- **Protection des données** : Seuls les utilisateurs authentifiés peuvent accéder aux fonctionnalités de réservation et aux tableaux de bord.
- **Gestion des sessions** : Sécurisation des interactions utilisateurs et journalisation des actions importantes dans la table *logs* pour garantir la traçabilité.

4.2. Système de réservation et gestion des conflits (Overlapping).

Le cœur de l'application permet de gérer les demandes de ressources en temps réel:

- **Processus de réservation** : L'utilisateur interne soumet une demande précisant la période, les ressources souhaitées et une justification.
- **Vérification automatique** : Un algorithme vérifie la disponibilité des ressources avant toute validation pour éviter le chevauchement (overlapping) de deux réservations sur un même créneau.
- **États de la demande** : Chaque demande suit un cycle de vie précis : En attente, Approuvée, Refusée, Active ou Terminée.

4.3. Gestion du parc matériel (Serveurs, VM, Stockage).

L'application propose un catalogue complet des ressources informatiques :

- **Catégorisation** : Les ressources sont classées par type : serveurs physiques, machines virtuelles, baies de stockage et équipements réseau.

- **Fiche technique détaillée** : Chaque ressource affiche ses caractéristiques propres : CPU, RAM, OS, capacité de stockage et bande passante.
- **Suivi opérationnel** : Le système permet de suivre l'état de chaque matériel (disponible, en maintenance, indisponible) et son historique d'utilisation.

4.4. Système de notifications et suivi des demandes.

Pour assurer une communication fluide entre les acteurs, un système de suivi a été mis en place :

- **Notifications automatiques** : Les utilisateurs reçoivent des alertes par messagerie interne concernant les réponses à leurs demandes, les expirations de réservation ou les conflits.
- **Tableau de bord** : Un espace dédié permet de visualiser les statistiques graphiques du taux d'occupation et l'historique des réservations filtré par date ou par état.
- **Signalement d'incidents** : Un module permet aux utilisateurs de rapporter tout problème technique directement aux responsables.

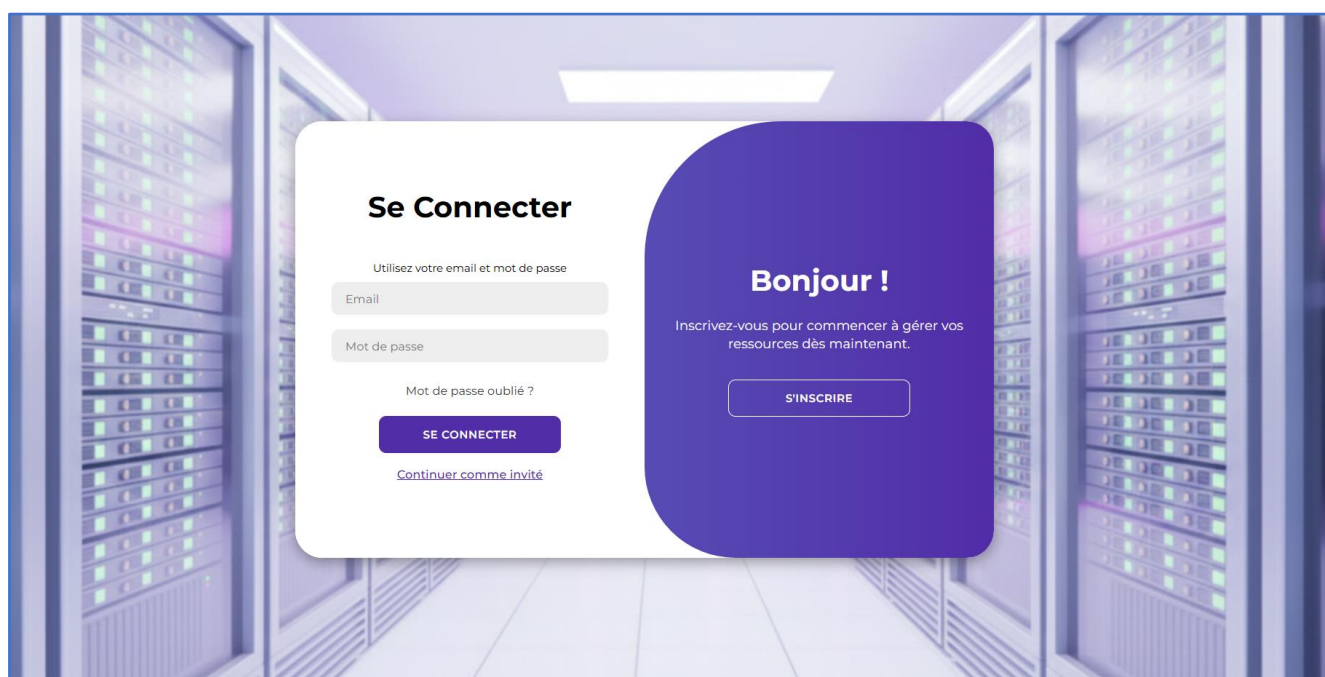
5. Réalisation et Démonstration (Captures d'écran)

Cette section présente l'aboutissement visuel du projet. L'interface a été entièrement réalisée en **CSS3 natif**, garantissant une légèreté et une rapidité d'affichage optimales sans dépendances externes.

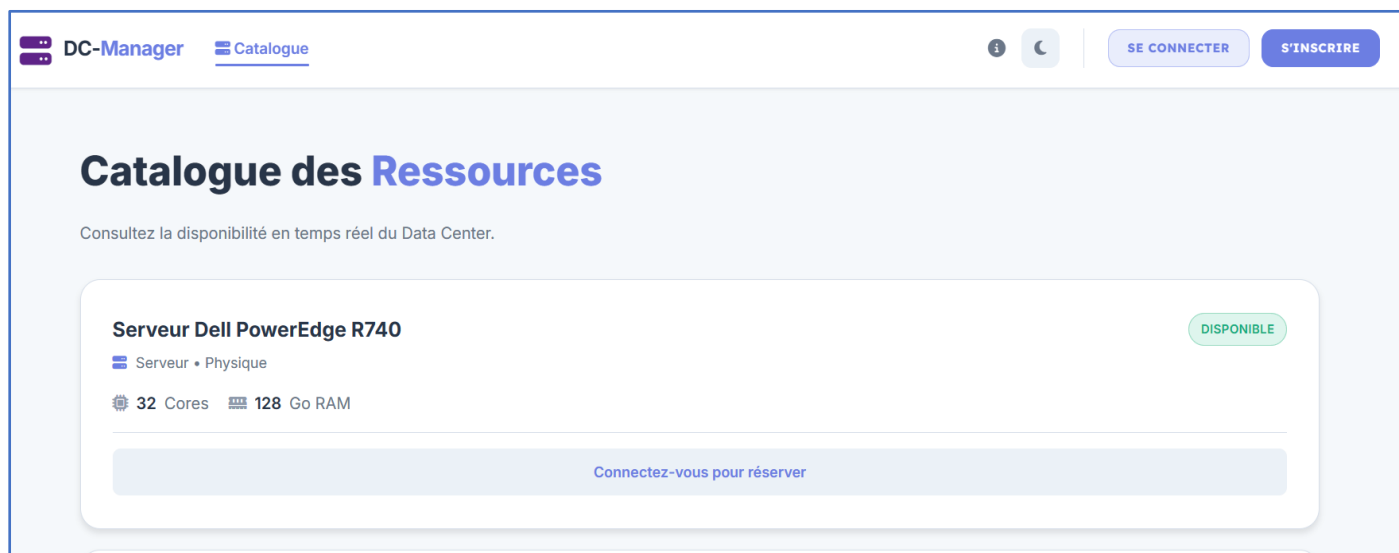
5.1. Interfaces utilisateurs et tableaux de bord :

Nous avons privilégié une ergonomie intuitive pour faciliter le processus de réservation.

- **L'Espace Public et Authentification** : La page de connexion sécurisée et la vue "**Invité**" permettant de consulter le catalogue et les règles d'utilisation avant toute inscription.



⇒ **Continuer comme invité :**



- **Le Tableau de Bord de l'Utilisateur Interne** : Centralise la recherche de ressources via des filtres techniques et le suivi en temps réel du statut des demandes (En attente, Approuvée, etc.).

⇒ **Dashboard :**

The screenshot shows the 'DC-Manager' dashboard. The top navigation bar includes 'DC-Manager', 'Dashboard' (active), 'Catalogue', and 'Mes Réservations'. On the right, there are notification, user, and theme icons, and a user profile for 'Ingénieur Réseau'. The main content area is titled 'Dashboard' and includes a welcome message. It features three key metrics: 'TAUX D'OCCUPATION' at 0%, 'RESSOURCES TOTALES' at 1, and 'MES DEMANDES' at 0. Below these is a section for 'État de votre compte' with a welcome message and a note about 0 active reservations. The footer indicates '© 2026 - Gestion de Ressources Data Center IDAI'.

⇒ **Catalogue :**

The screenshot shows the 'Catalogue des Ressources' page. The top navigation bar is similar to the dashboard, with 'Catalogue' active. The main heading is 'Catalogue des Ressources' with a subtitle 'Consultez la disponibilité en temps réel du Data Center.' The featured resource is a 'Serveur Dell PowerEdge R740', categorized as 'Serveur • Physique', with specifications '32 Cores' and '128 Go RAM'. It is marked as 'DISPONIBLE'. Below the resource details is a blue 'Réserver' button. A section titled 'SIGNALER UN INCIDENT' contains a form with a 'Sujet de l'incident' field, a 'Décrivez le problème en détail...' text area, and an 'Envoyer le signalement' button.

⇒ Mes réservations :

DC-Manager

DashboardCatalogueMes Réservations

Ingénieur Réseau
Ingénieur Réseau

Mon Historique Data Center

Consultez et filtrez vos réservations passées et en cours.

RESSOURCE

ÉTAT

DATE

Ex: Serveur AI...Tous les étatsjj/mm/aaaa

FiltrerReset

Aucune réservation ne correspond à vos critères.

[Effacer les filtres](#)

⇒ Réserver une ressource :

Réserver une Ressource

Planifiez votre allocation de ressources Data Center en quelques secondes.

CHOISIR L'ÉQUIPEMENT

-- Liste des ressources disponibles --

DATE DE DÉBUTDATE DE FIN

jj/mm/aaaajj/mm/aaaa

JUSTIFICATION DU BESOIN

Expliquez pourquoi vous avez besoin de cette ressource...

Confirmer la réservationAnnuler

⇒ Notifications :

Historique des Notifications

Retrouvez ici toutes vos alertes récentes et passées.



Tout est calme ici

Aucune nouvelle notification pour le moment.

© 2026 - Gestion de Ressources Data Center IDAI

5.2. Gestion administrative et statistiques :

Le back-office permet une supervision complète du Data Center et une aide à la décision grâce aux données chiffrées.

- **Gestion du Parc et des Accès :** Interfaces permettant aux Responsables et Administrateurs de modifier l'état d'une ressource (Maintenance/Désactivé) ou de gérer les comptes utilisateurs.

DC-Manager

DashboardCatalogueMa GestionIncidentsUtilisateursLogs

Admin SystèmeAdministrateur

Gestion des Utilisateurs

Validez les nouveaux comptes et gérez les accès au Data Center.

3 MEMBRES AU TOTAL

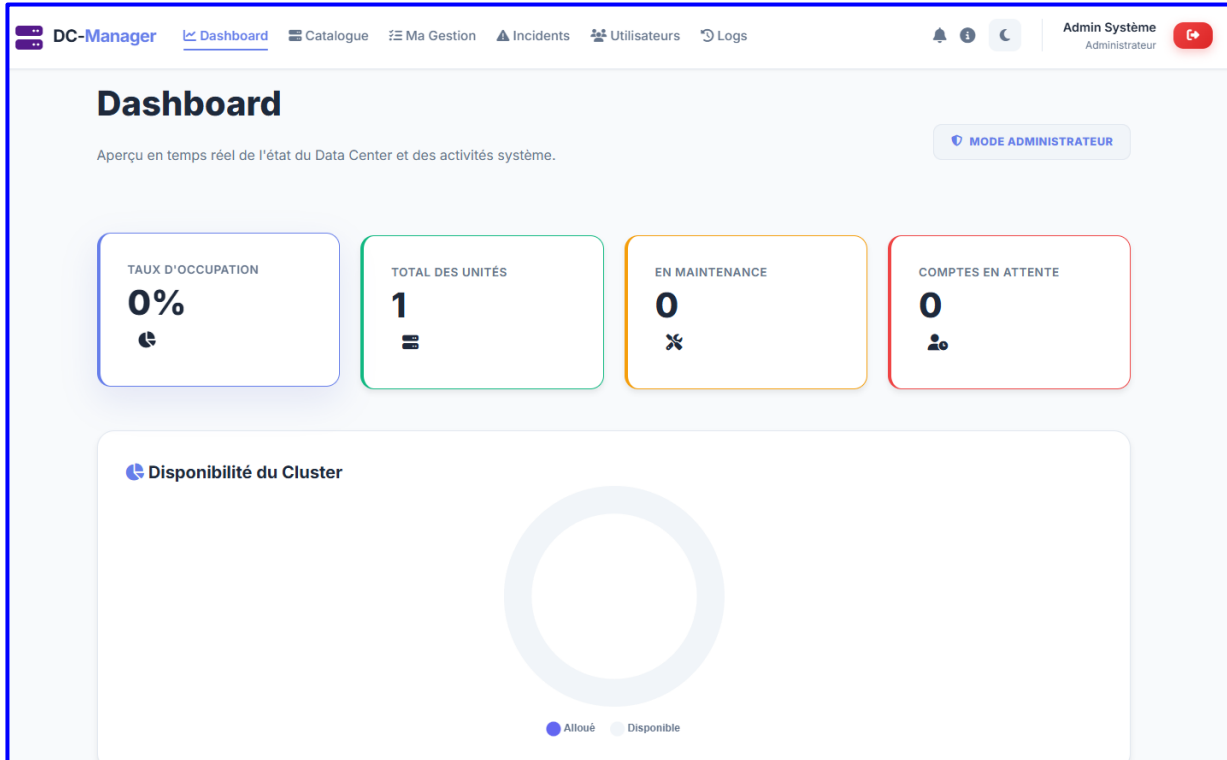
● Demandes d'ouverture de compte 0

Aucune demande en attente pour le moment.

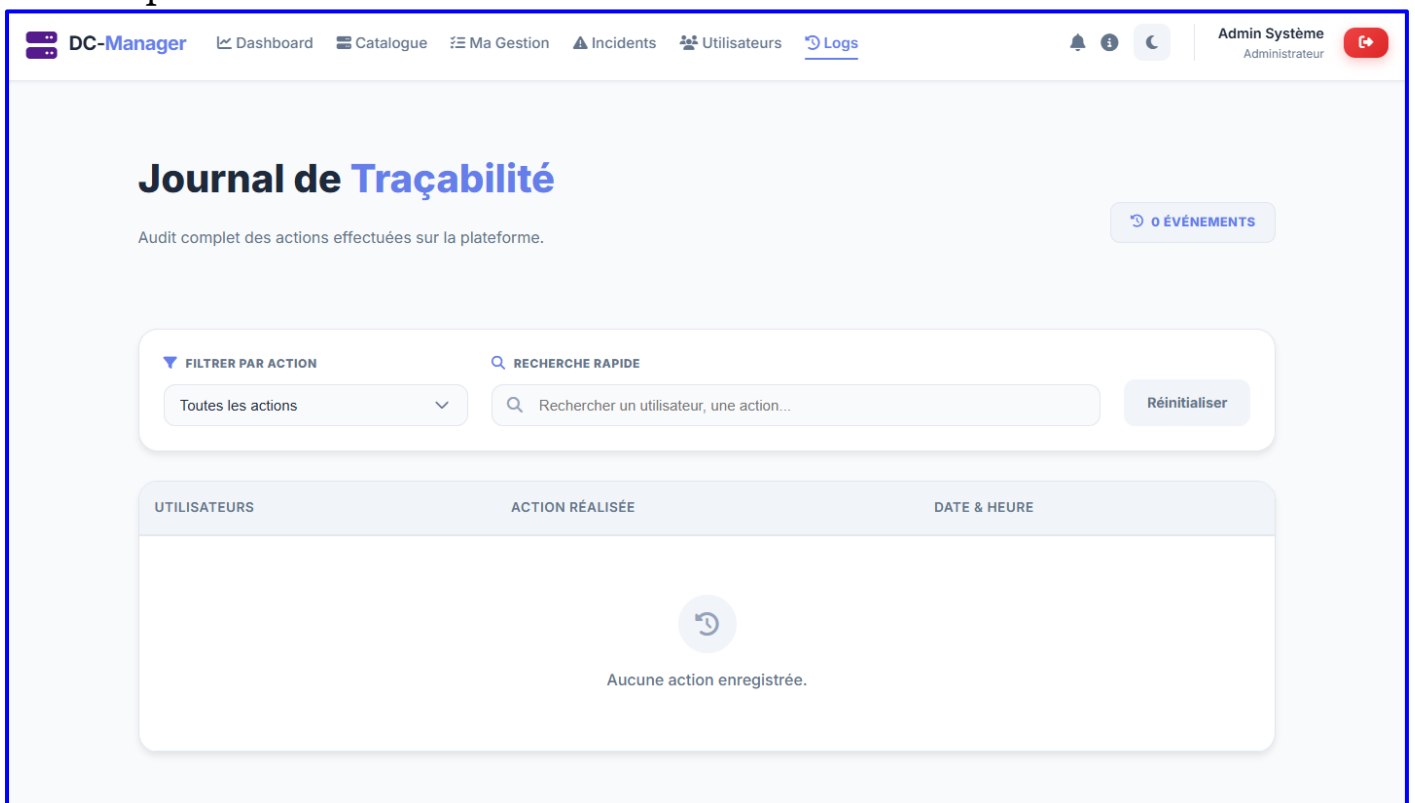
Gestion des accès existants

UTILISATEUR	RÔLE & PERMISSIONS	STATUT	ACTIONS
<div>R</div> <div>Responsable Tech</div> <div>responsable@datacenter.com</div>	<div>Responsable Tech</div> <div>Sauver</div>	<div>ACTIF</div>	<div>Révoquer l'accès</div>
<div>I</div> <div>Ingénieur Réseau</div> <div>user@datacenter.com</div>	<div>Ingénieur Réseau</div> <div>Sauver</div>	<div>ACTIF</div>	<div>Révoquer l'accès</div>

- **Module de Statistiques** : Visualisation graphique des données clés pour piloter le parc matériel.
 - ✓ Taux d'occupation global : Ratio ressources libres / ressources réservées.
 - ✓ Répartition des ressources : Volume de serveurs, VM et stockages utilisés.



- **Traçabilité** : Consultation de l'historique des *logs* pour l'audit des actions importantes.



6. Conclusion :

Le développement de cette application web de réservation des ressources d'un Data Center a permis de concevoir une solution robuste et sécurisée répondant aux besoins spécifiques de gestion d'une infrastructure complexe.

En s'appuyant sur l'architecture MVC du framework **Laravel** et sur une persistance de données gérée via **MySQL/phpMyAdmin** sous l'environnement **XAMPP**, ce projet a relevé des défis techniques majeurs tels que la gestion automatisée des conflits de réservation (overlapping) et le contrôle granulaire des accès par **Middlewares**.

L'approche choisie, privilégiant un **frontend natif en HTML5, CSS et JavaScript personnalisé**, a garanti une interface légère et performante sans dépendances externes.

Ce travail démontre ainsi l'efficacité d'une conception structurée pour transformer le suivi d'un parc matériel et les interactions entre les quatre profils d'utilisateurs en un système fluide, traçable et parfaitement opérationnel.