

RAPPORT COMPLET DE L'APPLICATION

Système de Gestion Intégré - Centre Diagnostic Libreville (CDL)

Date d'analyse : 2025

Version : 1.0

État : Application fonctionnelle avec modules de base

TABLE DES MATIÈRES

1. Vue d'ensemble
2. Architecture Technique
3. Modules Existant
4. Fonctionnalités Détaillées
5. Base de Données
6. API Backend
7. Modules Manquants

1. VUE D'ENSEMBLE

1.1 Description Générale

L'application est un Système de Gestion Intégré (SGI) pour le Centre Diagnostic Libreville, conçu pour gérer les opérations hospitalières incluant :

- Gestion des incidents (sécurité, entretien, technique)
- Gestion des visiteurs
- Planification des salles
- Gestion du parc biomédical
- Gestion des utilisateurs et rôles
- Tableaux de bord et KPIs

1.2 Technologies Utilisées

Frontend :

- React 18.3.1 avec TypeScript
- Vite 6.3.4
- Tailwind CSS 3.4.11
- Shadcn/ui (composants UI)
- React Query (@tanstack/react-query)

- React Router DOM 6.26.2
- Recharts 2.12.7 (graphiques)
- Zod 3.23.8 (validation)
- Date-fns 3.6.0 (gestion dates)

Backend :

- Node.js avec Express.js
- MySQL 2 (pool de connexions)
- JWT (authentification)
- Bcryptjs (hachage mots de passe)
- Multer (upload fichiers)
- Validation Joi

Base de Données :

- MySQL (WAMP/XAMPP)
- UTF8MB4 avec Unicode

2. ARCHITECTURE TECHNIQUE

2.1 Structure Frontend

...

src/

```
  └── components/      # Composants React organisés par domaine
      |   └── agent/      # Composants pour agents
      |   └── auth/       # Authentification
      |   └── biomedical/  # Gestion équipements biomédicaux
      |   └── dashboards/  # Tableaux de bord
      |   └── maintenance/ # Maintenance/Entretien
      |   └── planning/    # Planification salles
      |   └── qhse/        # Composants QHSE
      |   └── security/   # Sécurité
      |   └── shared/     # Composants partagés
      |   └── technician/ # Techniciens
      |   └── ui/          # Composants UI Shadcn
    └── hooks/           # Hooks React personnalisés
    └── integrations/   # Intégrations API
    └── lib/             # Utilitaires et configurations
    └── pages/           # Pages principales
    └── types.ts         # Types TypeScript
    └── utils/           # Utilitaires (PDF, toast, etc.)
```

...

2.2 Structure Backend

...

backend/

```
|── server.js      # Serveur Express principal  
|── middlewares/  # Middlewares (validation, rate limiting)  
|   └── validation.js
```

```
|── uploads/       # Fichiers uploadés (images incidents)  
└── package.json
```

...

2.3 Base de Données

Tables principales :

- Utilisateurs (10 rôles différents)
- Incidents de sécurité/entretien/technique
- Registre des visiteurs
- Équipements biomédicaux

- Tâches de maintenance
- Salles de consultation
- Annuaire médecins
- Réservations de salles
- Tâches planifiées
- Notifications système

3. MODULES EXISTANTS

3.1 Module d'Authentification

Fonctionnalités :

- Connexion (signin) avec email/mot de passe
- Inscription (signup) avec validation
- Gestion des tokens JWT
- Rate limiting sur les tentatives de connexion
- Réinitialisation de mot de passe
- Gestion des sessions

Composants :

- `LoginPage.tsx`
- `ForgotPasswordDialog.tsx`
- `ResetPasswordDialog.tsx`

API Endpoints :

- `POST /api/auth/signin`
- `POST /api/auth/signup`
- `POST /api/auth/signout`
- `PUT /api/auth/password`

3.2 Module de Gestion des Utilisateurs

Fonctionnalités :

- Création de comptes utilisateurs
- Gestion des rôles (10 types)
- Gestion des permissions personnalisées
- Réinitialisation de mots de passe
- Suppression d'utilisateurs

- Recherche et filtrage d'utilisateurs

Composants :

- `UserManagement.tsx`
- `AddUserDialog.tsx`
- `EditUserPermissionsDialog.tsx`

Rôles disponibles :

1. Super Admin
2. Superviseur QHSE
3. Superviseur Agent de Sécurité
4. Agent de Sécurité
5. Agent d'Entretien
6. Secrétaire
7. Médecin
8. Technicien

API Endpoints :

- `GET /api/users`
- `POST /api/users`
- `PUT /api/users/:id`

- `DELETE /api/users/:id`
- `PUT /api/users/:id/permissions`

3.3 Module de Gestion des Incidents

Fonctionnalités :

- Signalement d'incidents (sécurité, entretien, technique)
- Upload de photos (max 10MB)
- Gestion des statuts (nouveau, en cours, traité, résolu, attente)
- Gestion des priorités (faible, moyenne, élevée, urgente)
- Assignation d'incidents aux agents
- Rapports d'intervention techniques
- Historique des incidents

Types d'incidents :

- Sécurité : agression, vol, intrusion, etc.
- Entretien : nettoyage, sanitaire, déchets, hygiène, etc.
- Technique : électrique, plomberie, climatisation, équipement médical, etc.

Composants :

paule winnya

- `SecurityIncidentsTable.tsx`
- `ReportSecurityIncidentForm.tsx`
- `QhseTicketsTable.tsx`
- `ReportProblemForm.tsx`
- `TechnicianInterventionsTable.tsx`
- `InterventionReportDialog.tsx`

API Endpoints :

- `GET /api/incidents`
- `POST /api/incidents`
- `PUT /api/incidents/:id/status`
- `PUT /api/incidents/:id/assign`
- `PUT /api/incidents/:id/unassign`
- `POST /api/incidents/:id/report`

--

3.4 Module de Gestion des Visiteurs

Fonctionnalités :

- Enregistrement d'entrée de visiteurs

- Enregistrement de sortie
- Gestion des documents d'identité
- Raisons de visite
- Destinations dans l'établissement
- Personne à rencontrer
- Historique complet

Composants :

- `VisitorLog.tsx`

API Endpoints :

- `GET /api/visitors`
- `POST /api/visitors`
- `PUT /api/visitors/:id/signout`

3.5 Module de Planification des Salles

Fonctionnalités :

- Création de réservations
- Modification de réservations

- Annulation de réservations
- Vue matrice (grille horaire)
- Vue liste (par salle)
- Vue globale des salles
- Alertes visuelles (expiration imminente)
- Validation PIN pour médecins
- Démarrer/Terminer consultations

Composants :

- `RoomSchedule.tsx`
- `RoomScheduleMatrix.tsx`
- `GlobalRoomOverview.tsx`
- `AddBookingDialog.tsx`
- `EditBookingDialog.tsx`
- `BookingDetailsDialog.tsx`
- `PinValidationDialog.tsx`

API Endpoints :

- `GET /api/rooms`
- `GET /api/bookings`
- `POST /api/bookings`

- `PUT /api/bookings/:id`
- `DELETE /api/bookings/:id`
- `PUT /api/bookings/:id/start`
- `PUT /api/bookings/:id/end`

3.6 Module Biomédical

Fonctionnalités :

- Gestion du parc d'équipements biomédicaux
- Ajout/modification d'équipements
- Suivi des statuts (opérationnel, en maintenance, hors service)
- Planification des maintenances préventives
- Historique des maintenances
- Suivi des dates de maintenance

Composants :

- `EquipmentList.tsx`
- `AddEquipmentDialog.tsx`
- `MaintenanceSchedule.tsx`
- `ScheduleMaintenanceDialog.tsx`

API Endpoints :

- `GET /api/biomedical-equipment`
- `POST /api/biomedical-equipment`
- `PUT /api/biomedical-equipment/:id`
- `GET /api/maintenance-tasks`
- `POST /api/maintenance-tasks`

3.7 Module de Planification des Tâches

Fonctionnalités :

- Création de tâches planifiées
- Assignation aux agents
- Suivi des statuts (à faire, en cours, terminée, annulée)
- Dates d'échéance
- Tâches récurrentes
- Historique

Composants :

- `TaskPlanning.tsx`

- `CreateTaskDialog.tsx`
- `MyTasks.tsx`

API Endpoints :

- `GET /api/planned-tasks`
- `POST /api/planned-tasks`
- `PUT /api/planned-tasks/:id`
- `DELETE /api/planned-tasks/:id`

3.8 Module des Tableaux de Bord

Fonctionnalités :

- Dashboard Superadmin (vue globale)
- Dashboard Sécurité
- Dashboard Entretien
- Dashboard Technicien
- Dashboard QHSE
- KPIs (Indicateurs de performance)
- Graphiques et statistiques
- Métriques en temps réel

Composants :

- `SuperadminDashboard.tsx`
- `SecurityDashboard.tsx`
- `KpiDashboard.tsx`
- `DashboardCard.tsx`

Graphiques disponibles :

- Activité des 7 derniers jours (ligne)
- Répartition des incidents par statut (camembert)
- Indicateurs par service (camembert)

3.9 Module Notifications

Fonctionnalités :

- Notifications en temps réel
- Badge de notifications non lues
- Marquage comme lu
- Liens vers les éléments concernés
- Historique des notifications



Composants :

- `NotificationBell.tsx`

API Endpoints :

- `GET /api/notifications`
- `POST /api/notifications`
- `PUT /api/notifications/mark-read`

3.10 Module Annuaire Médecins

****Fonctionnalités :****

- Liste des médecins
- Spécialités
- Statuts (disponible, occupé, absent)

Composants :

- `DoctorList.tsx`

API Endpoints :

- `GET /api/doctors`

paule winnya

- `POST /api/doctors`
- `PUT /api/doctors/:id`

3.11 Module Informations Personnelles

Fonctionnalités :

- Affichage des informations utilisateur
- Modification du profil
- Changement de mot de passe

Composants :

- `PersonallInfo.tsx`

4. BASE DE DONNÉES

4.1 Tables Existantes

Tables
Utilisateurs
Incidents
Visiteurs
Équipements biomédical
Maintenances
Salles
Médecins
Réservations
Tâches planifiées
Notifications

5. MODULES MANQUANTS

5.1. Gestion Documentaire (GED QHSE)

- Création, validation, archivage de documents
- Gestion des versions et révisions
- Contrôle d'accès par profil
- Upload de fichiers

5.1.2. Audits & Inspections

- Programmation d'audits (interne, externe, certification)
- Checklists digitales
- Suivi des non-conformités
- Plan d'action automatique

5.1.3. Formations & Compétences

- Suivi des formations QHSE
- Habilitations et certificats
- Alertes sur échéances
- Vision des compétences par employé

5.1.4. Suivi des Déchets Médicaux

- Enregistrement et pesée
- Filière d'élimination
- Traçabilité complète
- Registres numériques

5.1.5. Suivi Stérilisation & Linge

- Registres numériques des cycles

- Traçabilité complète
- Alertes anomalies
- Suivi du linge

5.1.6. Gestion des Risques

- Identification des risques
- Évaluation (probabilité × sévérité)
- Plan d'action de traitement
- Calcul automatique du niveau de risque

5.1.7. Reporting & Exportation

- Génération automatique de rapports
- Formats prévus : PDF, Excel, Word (structure prête)

5.1.8. Amélioration Incidents (CAPA)

- Actions correctives et préventives
- Analyse de cause racine
- Suivi de récurrence
- Statut CAPA avec dates d'échéance

CONCLUSION

L'application actuelle dispose d'une base solide avec les modules essentiels pour la gestion opérationnelle d'un établissement médical. Cependant, il manque des modules pour une solution complète de gestion intégrée .