**📘 Cahier des charges : Application Web de Gestion de Maintenance Industrielle**

**1. 🎯 Objectif principal du projet**

Développer une application web performante et sécurisée pour centraliser la gestion des interventions de maintenance, permettant aux techniciens, administrateurs et superviseurs d’assurer le suivi des équipements, la traçabilité des opérations, et la visualisation des statistiques de performance.

**2. 👥 Utilisateurs & rôles**

| **Rôle** | **Accès & fonctions principales** |
| --- | --- |
| **Technicien** | - Créer des rapports de dépannage - Joindre des fichiers (photos, vidéos, etc.) - Suivre ses interventions |
| **Administrateur** | - Gérer les utilisateurs et les droits - Lire et valider les rapports - Accéder à tout l’historique |
| **Direction / Superviseur** | - Visualiser les statistiques - Exporter les rapports - Suivre les performances du parc |

**3. 🧩 Fonctionnalités clés**

**🔒 Authentification & sécurité**

* Connexion sécurisée via JWT
* Gestion fine des rôles et permissions

**📊 Dashboard intelligent**

* Vue synthétique : machines en panne, interventions en cours, MTTR/MTBF, pièces critiques

**🛠️ Gestion des interventions**

* Création et affectation des demandes
* Saisie des diagnostics, solutions, temps passé
* Ajout de fichiers (images, vidéos, PDF, etc.)
* Suivi par statut : Non traité → En cours → Résolu → Rejeté

**🏭 Gestion du parc machines**

* Fiche technique pour chaque machine
* Historique complet des interventions
* Photos avant/après

**📦 Gestion des stocks**

* Suivi des pièces utilisées
* Alertes sur seuil critique
* Référencement par fournisseur, prix, disponibilité

**📤 Export & Reporting**

* PDF/Excel des rapports journaliers, hebdo, mensuels
* Rapports par technicien, machine, période

**📈 Statistiques & Visualisation**

* Graphiques dynamiques (Recharts, Chart.js)
* Indicateurs : temps moyen de réparation, taux de récurrence, performances par équipe

**🔔 Notifications**

* Alertes push / email en cas de nouvelles affectations, clôtures ou urgences

**📚 Historique & audit**

* Suivi des actions utilisateur
* Logs d’accès et de modification

**📱 Interface responsive & PWA**

* Accessible via mobile, tablette, et desktop
* Application optimisée pour un usage en usine ou sur le terrain

**4. 🛠️ Architecture technique & codage**

**📁 Structure du projet**

/maintenance-system

├── backend/ # Express + Sequelize + PostgreSQL

├── frontend/ # React + Material UI

├── docker-compose.yml # Conteneurisation

**⚙️ Backend**

* Express.js pour les routes sécurisées
* PostgreSQL local ou distant (configurable)
* ORM : Sequelize (modèles, migrations)
* Upload de fichiers via Multer
* Validation avec Joi
* Swagger pour documentation

**🖥️ Frontend**

* React + Material UI
* Axios pour appels API
* Formulaires réactifs avec validation
* Dashboard graphique intégré

**5. 🧪 Tests et sécurité**

* Tests unitaires avec Jest (backend) et Testing Library (frontend)
* Protection contre XSS, CSRF, bruteforce, injections
* Logs d’erreurs avec Winston

**6. 🚀 Déploiement**

* Déploiement via Docker / Docker Compose
* CI/CD GitHub Actions pour automatiser les mises à jour
* Hébergement : VPS ou Vercel (selon budget/contraintes)

**7. 📂 Livrables attendus**

* Code source complet (GitHub)
* Fichiers Docker pour déploiement
* Scripts d’installation + migration base PostgreSQL
* Documentation technique + Swagger
* Guide utilisateur
* Exemples de rapports PDF/Excel

**8. 💎 Valeur ajoutée**

* Optimisation du temps d’intervention
* Meilleure coordination des équipes techniques
* Centralisation de l’historique des interventions
* Base solide pour extension future (capteurs IoT, maintenance prédictive)