ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

StockCare

Plateforme Intelligente de Gestion de Stock Pharmaceutique

Réalisé par :

Youlakou Marouane
Nassrellah Naji

Encadré par :

Mme. Soumaya Ounacer

Année Universitaire 2025 - 2026

Remerciements

Nous tenons à exprimer nos sincères remerciements à toutes les personnes qui ont contribué à la réalisation de ce projet de fin d'année.

Nous adressons nos remerciements particuliers à notre encadrante, Madame Soumaya Ounacer, pour son encadrement, sa disponibilité, ses conseils précieux et son soutien tout au long de ce travail.

Nous remercions également l'ensemble des enseignants et le personnel de l'EMSI pour la qualité de la formation dispensée et pour l'accompagnement dont nous avons bénéficié durant notre parcours académique.

Enfin, nous remercions nos camarades ainsi que nos familles pour leurs encouragements et leur soutien moral tout au long de la réalisation de ce projet.

Résumé

Ce rapport détaille la conception et la réalisation de **StockCare**, une plateforme web innovante dédiée à la gestion des stocks pharmaceutiques en milieu hospitalier. Face aux problématiques de rupture de stock, de péremption coûteuse et de manque de traçabilité, ce projet propose une solution numérique complète et sécurisée.

L'application permet de piloter l'ensemble du flux logistique, de la réception des commandes à la délivrance des médicaments, en intégrant des règles de gestion intelligentes comme le FEFO (*First Expired, First Out*). Elle offre également des tableaux de bord décisionnels pour une vision en temps réel de l'état des stocks.

D'un point de vue technique, StockCare repose sur une architecture moderne et robuste associant **NestJS** (Backend) et **Next.js** (Frontend), le tout conteneurisé avec **Docker** pour faciliter le déploiement. Ce projet a été conduit selon la méthodologie agile **Scrum**, garantissant une adéquation continue avec les besoins utilisateurs.

Mots-clés : Gestion de stock, Pharmacie Hospitalière, Traçabilité, FEFO, NestJS, Next.js, Docker, Scrum, Business Intelligence.

Table des matières

Table des figures

Présentation de l'École EMSI

À propos de l'EMSI

L'École Marocaine des Sciences de l'Ingénieur (EMSI), membre du réseau *Honoris United Universities*, est un acteur majeur de l'enseignement supérieur privé au Maroc. Fondée en 1986, elle s'est donnée pour mission de former des ingénieurs hautement qualifiés, capables de répondre aux défis technologiques et économiques actuels.

Notre Campus et Nos Valeurs

L'école se distingue par son approche pédagogique axée sur l'innovation, la recherche scientifique et l'ouverture sur le monde de l'entreprise. À travers ses différents campus, l'EMSI offre un cadre d'apprentissage stimulant, favorisant l'épanouissement personnel et professionnel de ses futurs ingénieurs.

Ce projet de fin d'études s'inscrit pleinement dans la philosophie de l'EMSI, visant à concrétiser les acquis théoriques par la réalisation d'une solution technologique innovante et utile à la société.

Introduction Générale

L'ère numérique actuelle impose une transformation profonde dans tous les secteurs d'activité, et le domaine de la santé ne fait pas exception. La digitalisation des processus hospitaliers est devenue une nécessité impérieuse pour garantir une qualité de soins optimale, une sécurité accrue pour les patients et une efficacité opérationnelle durable. Au cœur de cette dynamique, la gestion logistique des produits pharmaceutiques représente un enjeu stratégique majeur. En effet, la disponibilité des médicaments, leur traçabilité et le contrôle de leur validité sont des paramètres critiques qui impactent directement la vie des patients et l'économie des établissements de santé.

Cependant, malgré les avancées technologiques, de nombreuses structures, notamment les pharmacies hospitalières de taille intermédiaire, continuent de s'appuyer sur des méthodes de gestion traditionnelles, souvent manuelles ou peu intégrées. Ces pratiques, bien qu'historiques, montrent aujourd'hui leurs limites face à la complexité croissante des flux logistiques et aux exigences réglementaires de plus en plus strictes. Les erreurs de saisie, les ruptures de stock inopinées et les pertes liées à la péremption des produits constituent des défis quotidiens qui fragilisent le système de santé.

C'est dans ce contexte que s'inscrit le projet **StockCare**. Il ne s'agit pas simplement de développer un logiciel de gestion, mais de concevoir une véritable solution intelligente capable de moderniser l'approche logistique des pharmacies hospitalières. Notre ambition est de fournir un outil robuste, intuitif et sécurisé, permettant de passer d'une gestion réactive à un pilotage proactif des stocks.

Ce rapport retrace l'intégralité du cycle de vie de ce projet, depuis l'analyse des besoins sur le terrain jusqu'à la mise en production de la solution. Nous traverserons ensemble les phases de réflexion stratégique, de conception architecturale et de réalisation technique qui ont permis de donner naissance à StockCare. Ce document est le témoin de notre démarche d'ingénieur, alliant rigueur méthodologique et innovation technologique pour répondre à une problématique concrète et utile à la société.

Chapitre 1

Contexte et Méthodologie du Projet

1.1 Contexte du Projet

L'industrie de la santé connaît une mutation numérique sans précédent. Au cœur de cette transformation, la logistique hospitalière et pharmaceutique joue un rôle vital. La gestion des médicaments n'est pas une simple gestion de stock : elle touche directement à la sécurité des patients et à la conformité réglementaire.
Ce projet, intitulé **StockCare**, s'inscrit dans cette démarche de modernisation. Il vise à fournir aux établissements de santé un outil robuste pour maîtriser leurs flux logistiques.

1.2 Problématique

Malgré les avancées technologiques, de nombreuses pharmacies hospitalières dépendent encore de processus archaïques.

- **Comment garantir la traçabilité totale d'un médicament, de sa réception à sa dispensation ?**
- **Comment réduire le gaspillage dû aux péremptions, qui représente un coût financier et écologique majeur ?**
- **Comment sécuriser l'accès aux stocks sensibles ?**

C'est à ces questions que StockCare tente de répondre.

1.3 Objectifs et Périmètre

L'objectif est de développer une application web *Full Stack* permettant :

1. La gestion centralisée du référentiel produits.
2. Le suivi temps réel des niveaux de stock et des dates de péremption (DLU).
3. La génération d'alertes proactives pour le réapprovisionnement.
4. Une gestion fine des droits utilisateurs (RBAC).

1.4 Méthodologie de Travail

1.4.1 Méthodologie Scrum

Le projet a été piloté via la méthode **Scrum**, avec des itérations courtes (Sprints) permettant d'intégrer régulièrement les retours utilisateurs.

1.4.2 Planification (Gantt)

Le diagramme ci-dessous illustre le déroulement temporel du projet sur 12 semaines.

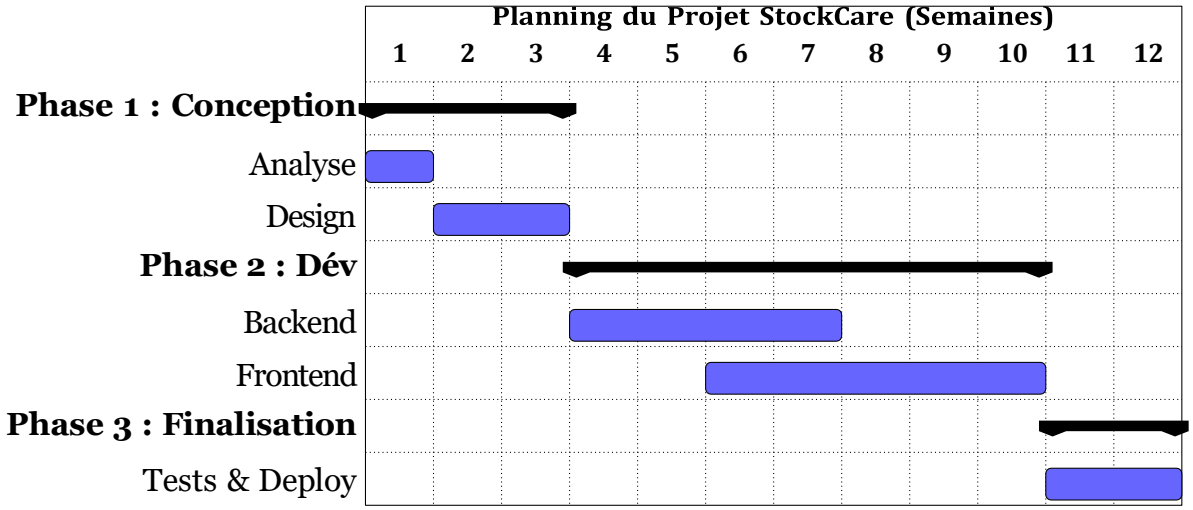


FIGURE 1.1 – Diagramme de Gantt détaillé

Conclusion

Au terme de ce chapitre introductif, nous avons cerné les enjeux majeurs liés à la gestion des stocks pharmaceutiques et défini notre cadre méthodologique. La définition claire du périmètre, des objectifs et de notre organisation Agile nous fournit une boussole pour la suite du projet. Il convient désormais de quitter la théorie pour le terrain, en analysant en profondeur les processus existants et les besoins réels des utilisateurs finaux, ce qui fera l'objet du chapitre suivant.

Chapitre 2

Analyse de l'Existant et Besoins

2.1 Introduction

La réussite d'un projet informatique dépend fondamentalement de la compréhension approfondie de l'environnement dans lequel il sera déployé. Ce chapitre d'analyse ne se contente pas de lister des fonctionnalités ; il vise à déconstruire les processus métier actuels, à diagnostiquer leurs défaillances, et à prescrire une solution numérique adaptée. Nous adopterons une approche structurée, allant de l'étude comparative des solutions existantes (Benchmarking) à la définition formelle des besoins via des personas utilisateurs.

2.2 Étude de l'Existant

2.2.1 Description des Processus Actuels

Dans la majorité des structures ciblées (pharmacies hospitalières de taille moyenne), le flux de travail est fragmenté :

1. **L'Inventaire** : Réalisé manuellement sur des cahiers ou des feuilles volantes. La mise à jour est asynchrone, créant un décalage permanent entre le stock réel et le stock perçu.
2. **La Commande** : Basée sur l'intuition du pharmacien ("Il me semble qu'on manque d'Amoxicilline") plutôt que sur des données objectives de consommation.
3. **La Vérification** : Lors de la réception, le contrôle des dates de péremption est souvent visuel et rapide, sans enregistrement numérique pour un suivi ultérieur.

2.2.2 Analyse Comparative (Benchmarking)

Nous avons comparé les outils actuellement utilisés sur le marché pour positionner notre solution :

Critère	Gestion Papier	Excel / Tableur	ERP Hospitalier
Coût	Négligeable	Faible	Très Élevé
Accessibilité	Locale uniquement	Locale / Cloud partiel	Réseau sécurisé
Traçabilité	Nulle	Faible (Modifiable)	Totale
Alertes	Aucune	Manuelle (Formules)	Automatique
Ergonomie	Simple	Moyenne	Complexe
Risque Erreur	Critique (Illisibilité)	Élevé (Saisie)	Faible

Conclusion du Benchmark : Il existe un vide sur le marché pour une solution intermédiaire : plus robuste qu'Excel, mais moins coûteuse et complexe qu'un ERP lourd type SAP ou Sage. C'est ici que **StockCare** se positionne.

2.3 Critique de l'Existant (Diagnostic SWOT)

Pour structurer notre analyse, nous avons réalisé une matrice SWOT (Forces, Faiblesses, Opportunités, Menaces) appliquée à la situation actuelle des pharmacies non-informatisées.

2.3.1 Faiblesses (Interne)

— **Perte de Temps** : La recherche physique d'un médicament ou d'un bon de livraison prend en moyenne 15 à 30 minutes par jour et par employé.
— **Pertes Financières** : Les produits périmés sont souvent découverts trop tard pour être retournés ou échangés, représentant une perte sèche de 5 à 10% du budget annuel.
— **Stress et Erreurs** : La charge mentale de devoir tout mémoriser augmente le risque d'erreur de délivrance.

2.3.2 Menaces (Externe)

— **Réglementation** : Les normes de traçabilité sanitaire se durcissent (sérialisation des médicaments). Les systèmes manuels ne sont plus conformes.
— **Exigence Patient** : La tolérance aux ruptures de stock de médicaments vitaux est nulle.

2.4 Identification des Acteurs (Personas)

Pour humaniser nos spécifications, nous avons défini trois "Personas" qui représentent nos utilisateurs types.

2.4.1 Mme. Ibtissam - La Pharmacienne Chef

— **Profil** : 45 ans, experte métier, peu à l'aise avec l'informatique complexe.
— **Objectif** : Garantir la disponibilité des soins et maîtriser son budget.
— **Frustration** : Jeter des médicaments périmés et passer ses soirées à faire des inventaires.
— **Attente** : Un tableau de bord simple qui lui dit quoi commander ce matin.

2.4.2 M. Rachid - Le Magasinier

— **Profil** : 28 ans, dynamique, utilise un smartphone toute la journée.
— **Objectif** : Ranger et sortir les stocks le plus vite possible.
— **Frustration** : Les interfaces PC lentes, écrire sur du papier qu'il perd ensuite.
— **Attente** : Une interface tactile, gros boutons, scan de code-barre.

2.4.3 M. Ahmed - L'Administrateur IT

— **Objectif** : Sécurité des données et facilité de maintenance.
— **Attente** : Une solution Dockerisée, facile à déployer et sauvegarder.

2.5 Spécification Détaillée des Besoins

2.5.1 Besoins Fonctionnels (Priorisés par MoSCoW)

MUST HAVE (Indispensable)

1. **Authentification RBAC** : Système de connexion sécurisé différenciant Admin / Pharmacien / Magasinier.
2. **Gestion des Entrées** : Saisie des BL (Bon de Livraison) avec numéro de lot et date de péremption obligatoire.
3. **Gestion des Sorties** : Décrémentation du stock avec règle FEFO (First Expired, First Out) automatique.
4. **Moteur de Recherche** : Recherche instantanée de produit par Nom, DCI ou Code-barre.

SHOULD HAVE (Important)

1. **Alertes Emails** : Notification automatique à J-30 et J-15 avant péremption.
2. **Dashboard Décisionnel** : Graphiques de valeur de stock et top consommation (Power BI).
3. **Seuil d'Alerte** : Coloration automatique des lignes de stock critique (< Stock Sécurité).

COULD HAVE (Confort)

1. **Export PDF/Excel** : Génération de bons de sortie imprimables.
2. **Historique d'Audit** : Logs complets de "Qui a fait quoi et quand".

2.5.2 Besoins Non-Fonctionnels (Qualité de Service)

— **Performance** : Le chargement des listes de produits doit se faire en moins de 1 seconde, même avec 10 000 références (Pagination côté serveur).
— **Sécurité** : Les mots de passe doivent être hachés (Bcrypt). Les échanges doivent être chiffrés (HTTPS).
— **Disponibilité** : L'application doit être résiliente aux pannes mineures (Architecture Conteneurisée).
— **Ergonomie** : Interface "Mobile First" pour permettre l'usage sur tablette dans le stock.

2.6 Conclusion

Cette phase d'analyse a permis de transformer un problème vague ("mieux gérer le stock") en une liste précise de fonctionnalités techniques et ergomiques. Nous savons désormais que le cœur de notre valeur ajoutée réside dans la **gestion proactive des péremptions** et la **simplicité d'usage** pour le personnel de terrain. Ces constats guideront directement nos choix d'architecture dans le chapitre suivant.

Chapitre 3

Conception et Modélisation

Introduction

Après avoir clairement défini "quoi" faire dans le chapitre précédent, nous allons maintenant nous concentrer sur le "comment". La phase de conception est le pont essentiel entre l'expression des besoins et l'implémentation technique. Dans ce chapitre, nous utiliserons le langage standard UML pour modéliser la structure statique et dynamique de notre application, et nous définirons l'architecture logicielle globale qui garantira la robustesse et l'évolutivité de StockCare.

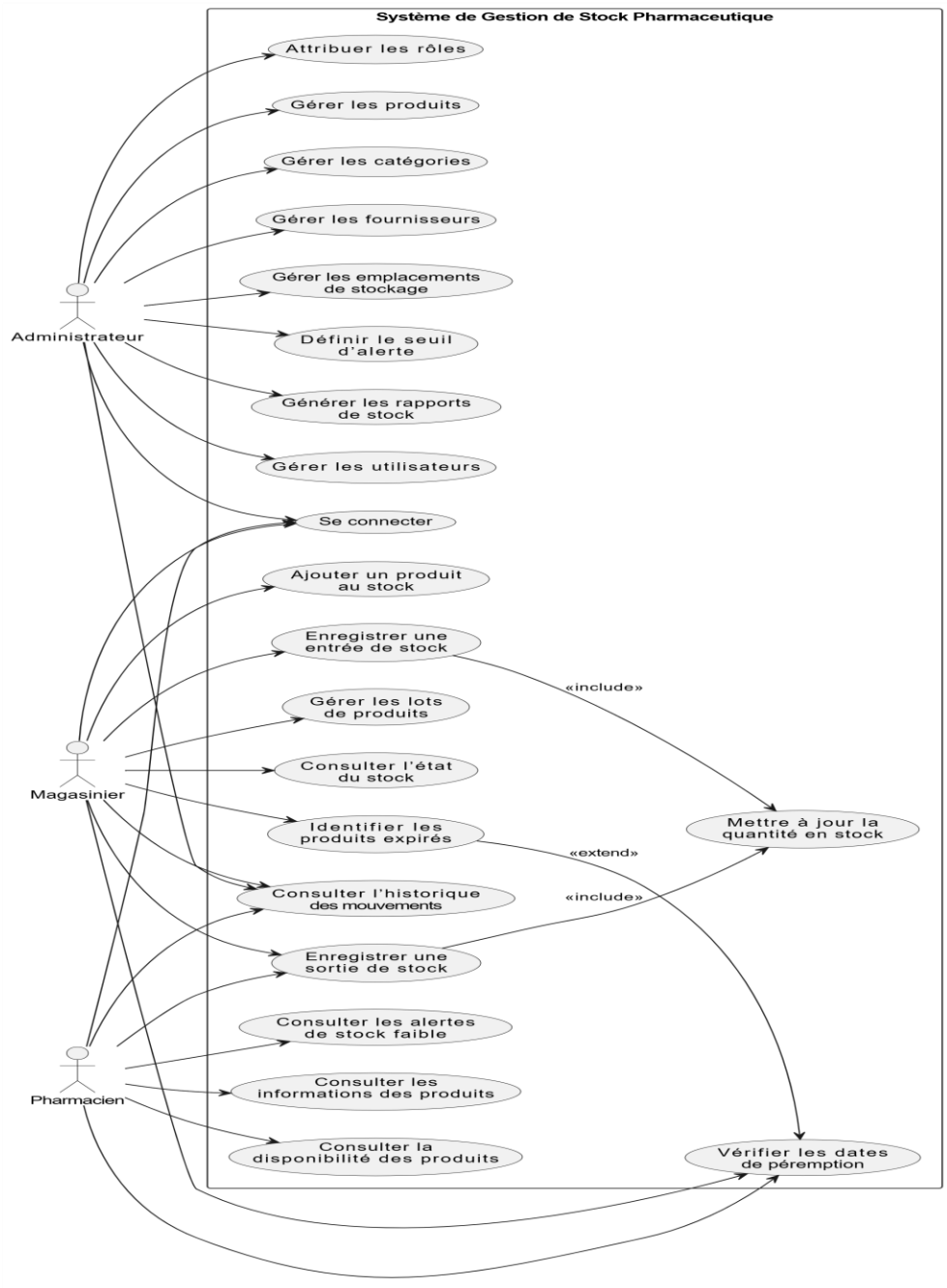
3.1 Architecture Globale

Nous avons opté pour une architecture **n-tiers** web, favorisant la séparation des préoccupations. Le Frontend (Client) communique avec le Backend (Serveur) via une API REST sécurisée.

3.2 Diagrammes UML

3.2.1 Diagramme de Cas d'Utilisation

FIGURE 3.1 – Diagramme de Cas d'Utilisation Global



3.2.2 Diagramme de Classes

Le cœur de notre système. Il définit la structure statique des données.

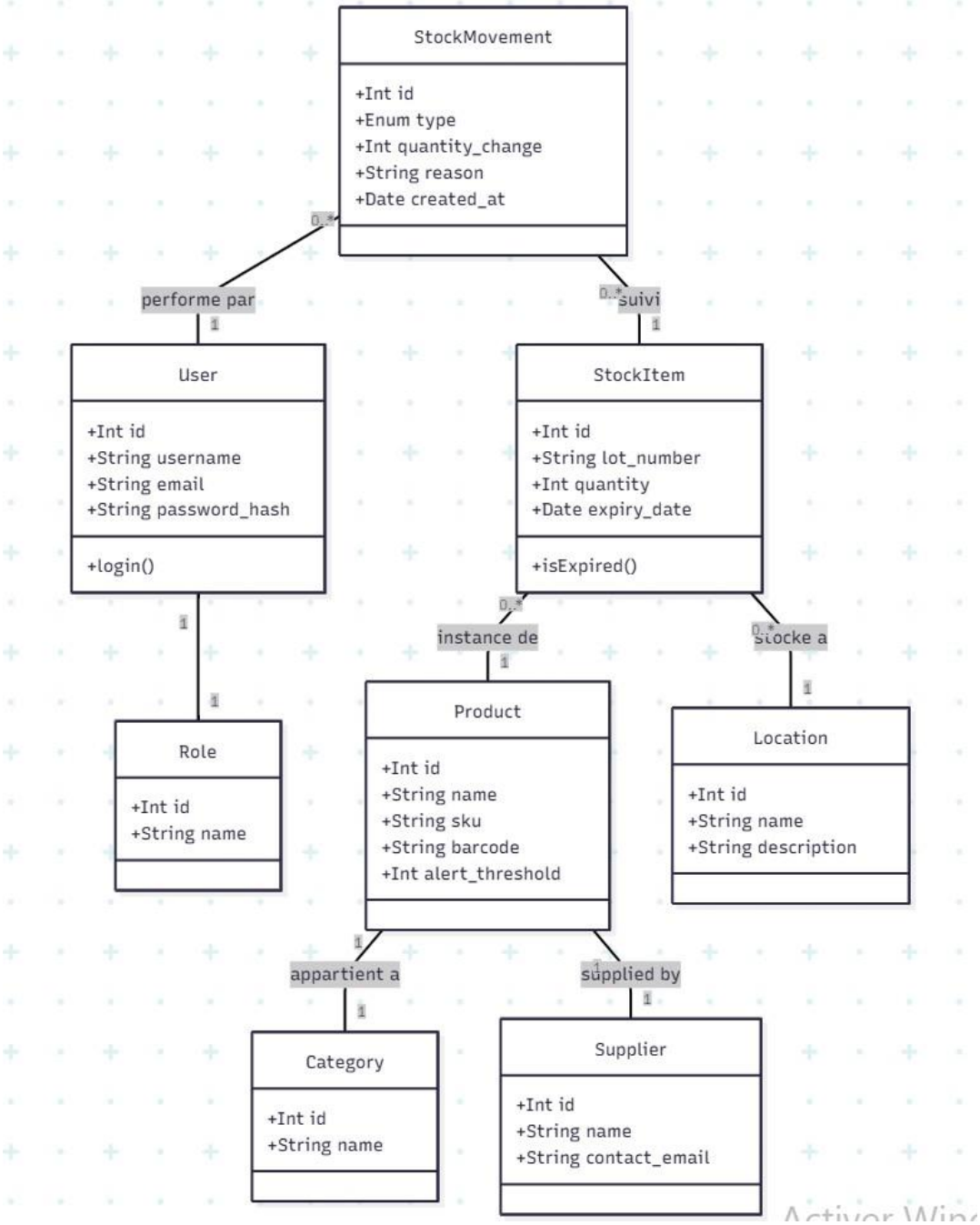


FIGURE 3.2 – Diagramme de Classes détaillé

3.2.3 Diagramme de Séquence : Entrée en Stock

Détaille le flux temporel lors de la réception d'une commande.

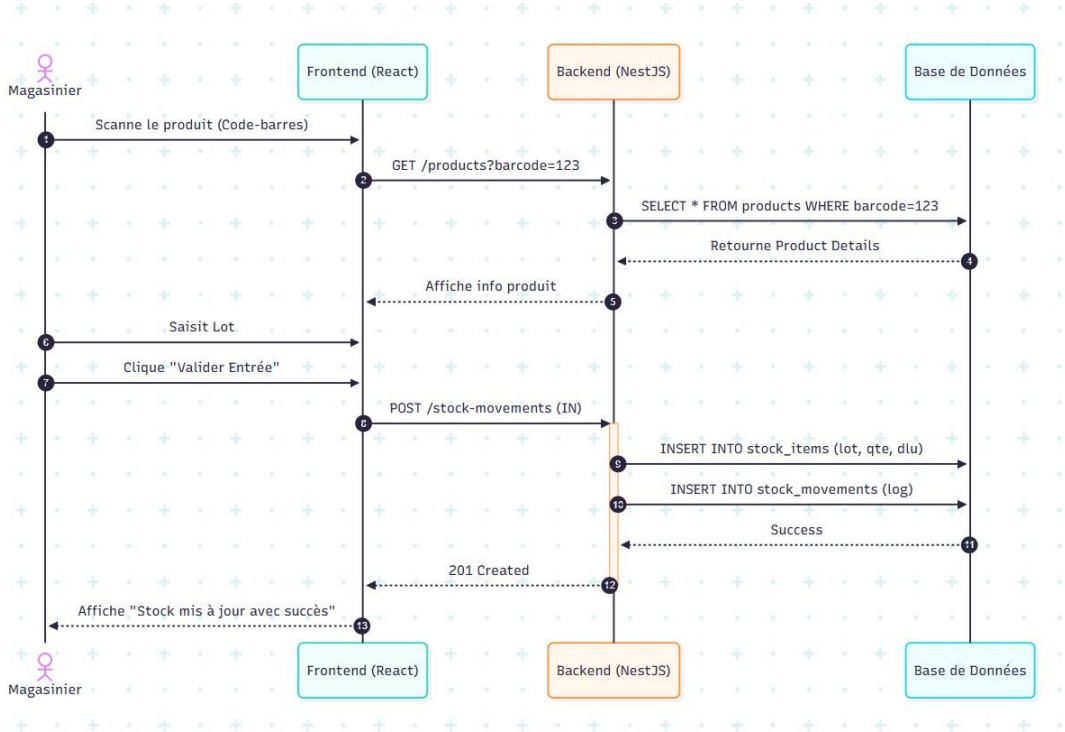


FIGURE 3.3 – Séquence : Réception de marchandises

Conclusion

La phase de conception et de modélisation nous a permis de visualiser le système dans son ensemble avant d'écrire la moindre ligne de code. Les diagrammes UML élaborés constituent désormais notre feuille de route technique. Avec cette vision claire de l'architecture et des interactions, nous sommes prêts à justifier le choix des outils technologiques qui donneront vie à ce modèle dans le chapitre suivant.

Chapitre 4

Environnement Technique et Outils

Introduction

Le choix de la stack technologique est une décision stratégique qui impacte directement la performance, la sécurité et la maintenabilité future du projet. Ce chapitre a pour but de présenter et de justifier l'ensemble des outils et technologies retenus pour la réalisation de StockCare. Nous détaillerons notre architecture PERN, ainsi que l'intégration d'outils modernes comme Docker pour le déploiement et Power BI pour l'analyse décisionnelle.

4.1 Services d'Analyse et Notifications

4.1.1 Business Intelligence : Power BI

Nous avons intégré une couche décisionnelle avec **Power BI**. Connecté à la base de données, il permet de générer des rapports dynamiques sur les stocks et de visualiser les tendances de consommation sur le long terme.

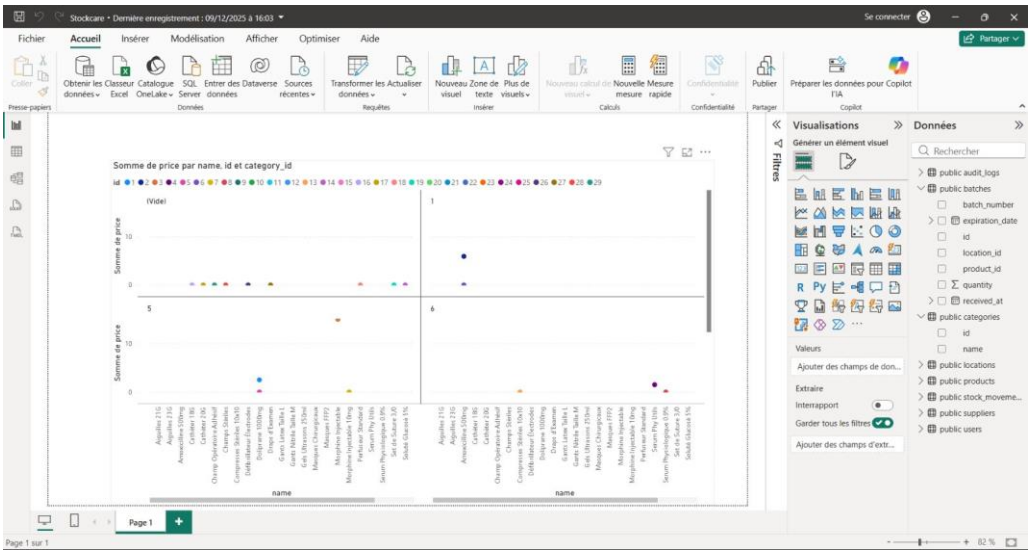


FIGURE 4.1 – Exemple de tableau de bord décisionnel Power BI

4.1.2 Gestion des Alertes : MailJS

Pour les notifications urgentes, nous utilisons **MailJS**. Ce service tiers nous permet d'envoyer des alertes email (stocks bas, péremption imminente) directement depuis l'application sans gérer de serveur SMTP complexe, garantissant une délivrabilité maximale.

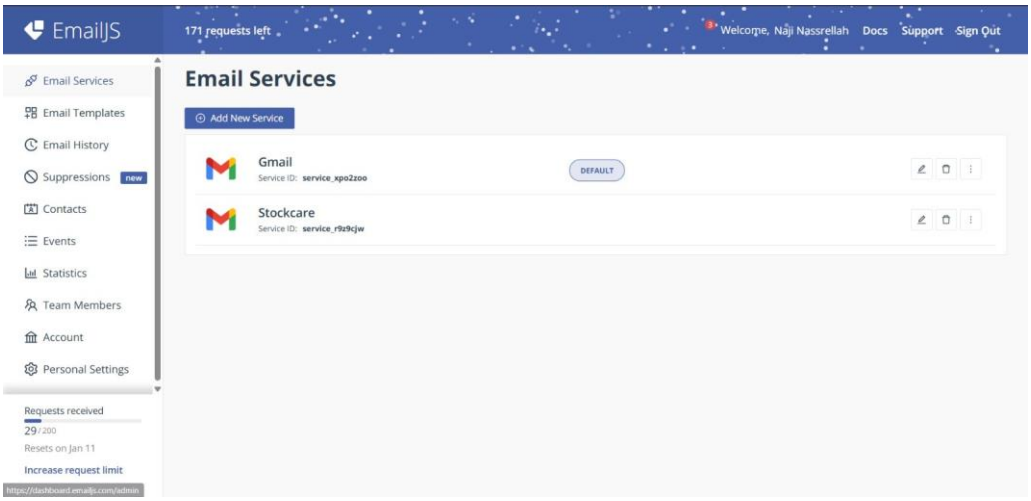


FIGURE 4.2 – Interface de configuration et exemple d'alerte MailJS

4.2 Infrastructure : Docker

L'application est entièrement conteneurisée avec **Docker**. Cela assure l'isolation des environnements (Backend, Frontend, BDD) et facilite le déploiement sur n'importe quel serveur, éliminant les problèmes de compatibilité ("It works on my machine").

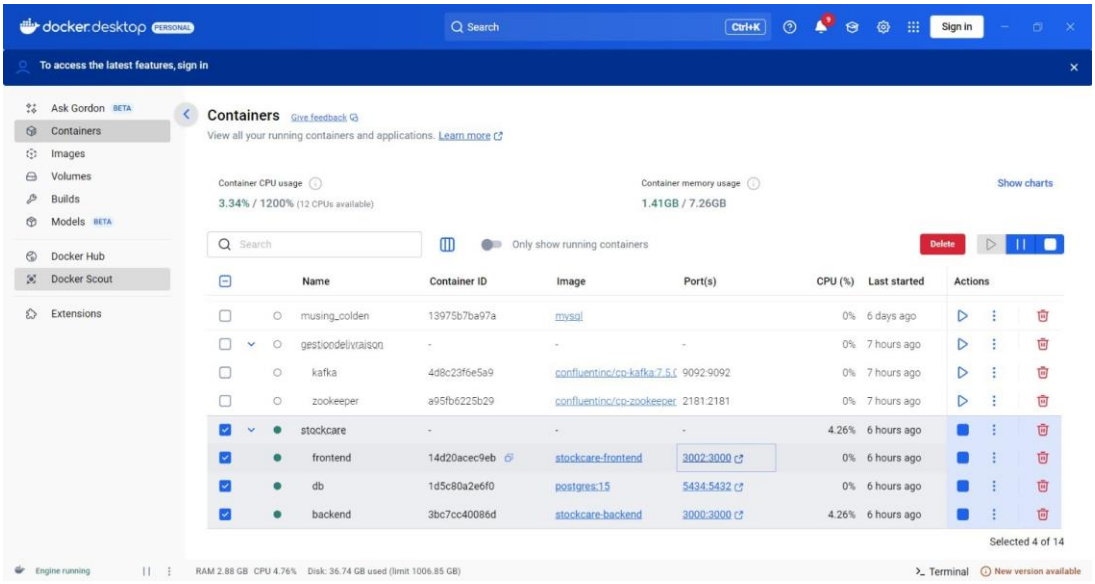


FIGURE 4.3 – Aperçu des conteneurs StockCare (Docker Desktop)

4.3 Cœur Applicatif

4.3.1 Le Backend : NestJS

NestJS impose une architecture modulaire stricte, idéale pour l'évolutivité. TypeORM sécurise nos interactions avec la base de données.

4.3.2 Le Frontend : Next.js

React 19 et Next.js 16 offrent une expérience utilisateur fluide et un référencement optimal grâce au rendu serveur (SSR).

Conclusion

L'environnement technique mis en place offre une base robuste, performante et évolutive, parfaitement adaptée aux exigences critiques du secteur médical. La combinaison de NestJS, Next.js, Docker et Power BI nous dote de tous les outils nécessaires pour aborder sereinement la phase de réalisation et transformer notre conception en un produit fonctionnel.

Chapitre 5

Réalisation de l'Application

Introduction

Ce chapitre est sans doute le plus concret de ce rapport, car il marque l'aboutissement de toute la réflexion précédente. Nous y présenterons le fruit de notre travail de développement : l'application StockCare. À travers des captures d'écran commentées, nous illustrerons le parcours utilisateur, l'ergonomie des interfaces et l'implémentation effective des fonctionnalités clés, de la gestion des stocks à la prédiction des anomalies.

5.1 Interfaces Utilisateurs

Cette section présente le parcours utilisateur à travers l'application.

5.1.1 Authentification

L'interface de connexion est la porte d'entrée sécurisée de l'application. Elle authentifie les utilisateurs via leur email et mot de passe, et les redirige vers le tableau de bord correspondant à leur rôle (Admin, Pharmacien, ou Magasinier) grâce au mécanisme RBAC. La sécurité est renforcée par un hachage des mots de passe en base de données.

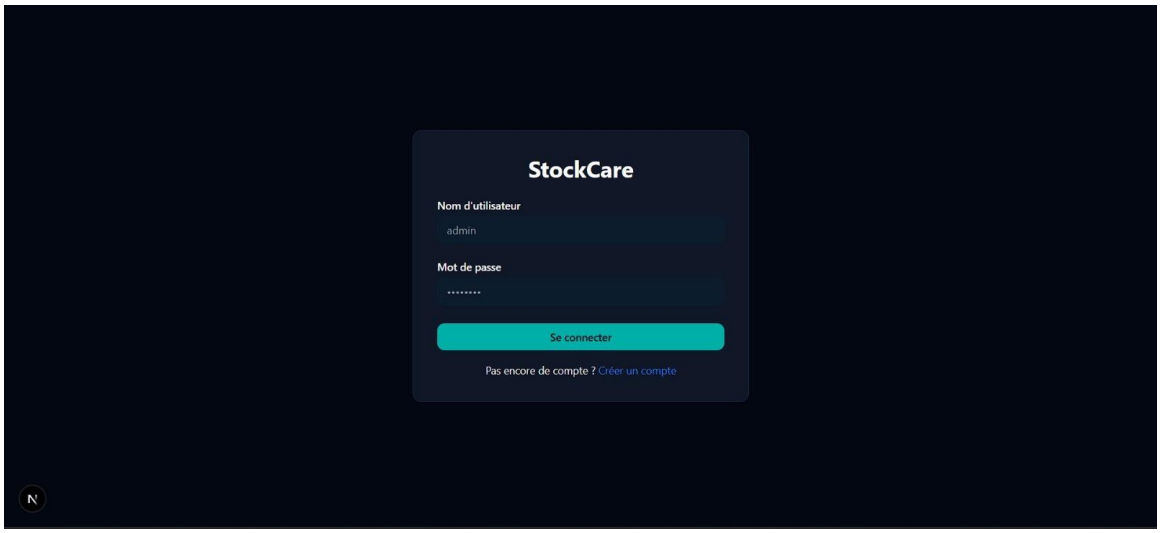


FIGURE 5.1 – Page de Connexion

5.1.2 Tableau de Bord (Dashboard)

Le tableau de bord est le centre de pilotage de StockCare. Il offre une vue d'ensemble immédiate sur l'état des stocks (valeur totale, produits critiques), les tâches en attente (commandes à valider) et les indicateurs clés de performance. C'est ici que l'utilisateur commence sa journée de travail et prend les décisions prioritaires.

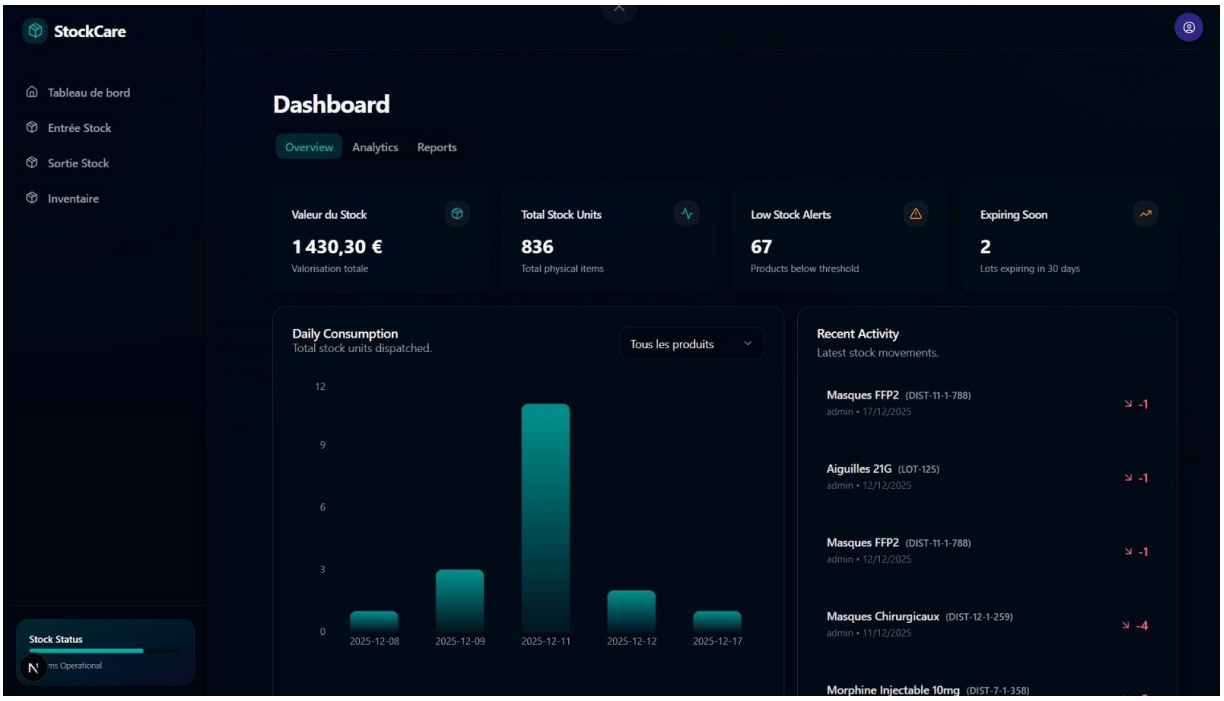


FIGURE 5.2 – Tableau de Bord Général

5.1.3 Gestion des Produits et Tiers

Pour assurer une traçabilité complète, StockCare propose des interfaces dédiées à la gestion des données référentielles. L'écran "Produits" permet d'ajouter et de modifier les fiches médicaments avec leurs caractéristiques détaillées (DCI, dosage, forme galénique). Les vues "Catégories" et "Fournisseurs" permettent d'organiser le catalogue produits et de gérer efficacement la relation avec les partenaires logistiques.

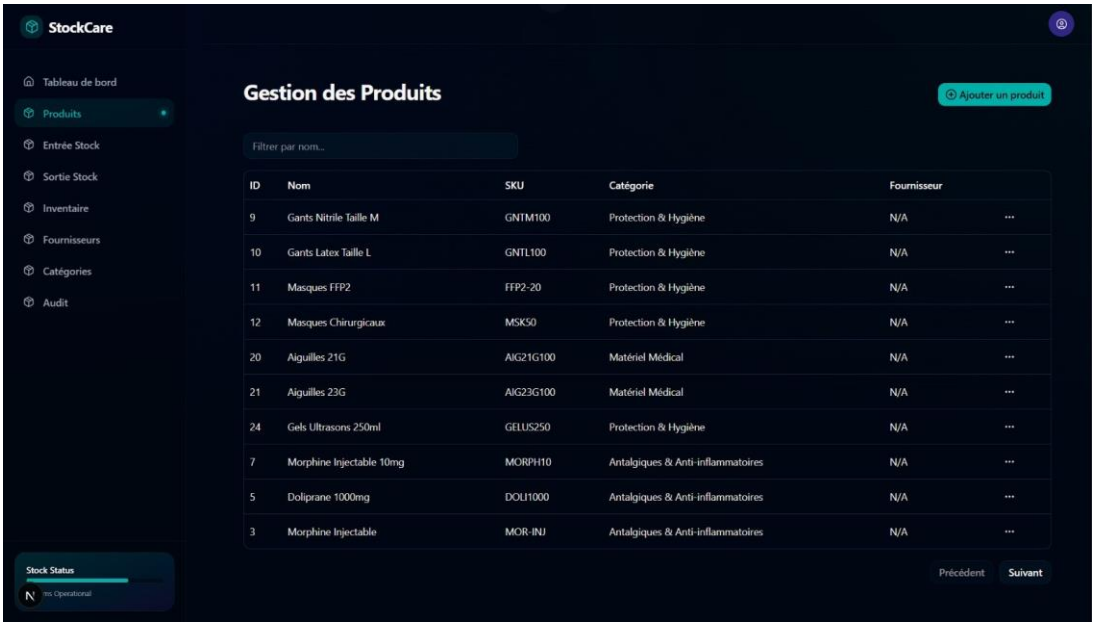


FIGURE 5.3 – Interface de la liste des produits

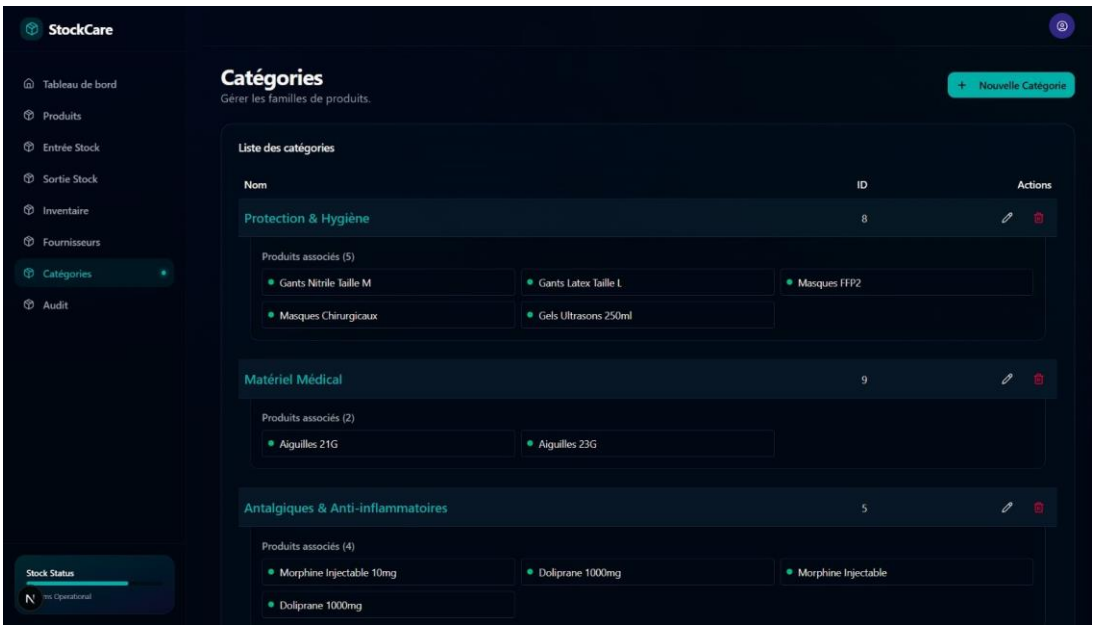


FIGURE 5.4 – Interface de gestion des catégories

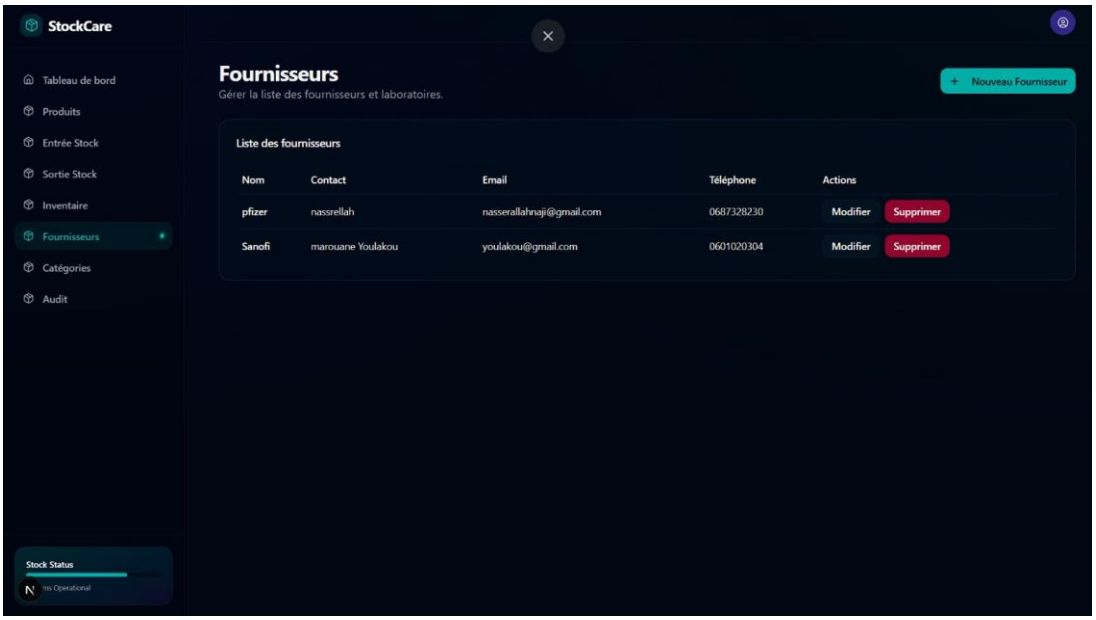


FIGURE 5.5 – Interface de la liste des fournisseurs

5.1.4 Gestion des Mouvements de Stock

La fluidité des opérations logistiques repose sur ces deux écrans essentiels. L'interface d'Entrée permet de saisir les réceptions en scannant les produits et en renseignant obligatoirement les lots et dates de péremption pour garantir la traçabilité. L'interface de Sortie, quant à elle, guide le magasinier pour déstocker les produits selon la règle FEFO (First Expired, First Out), minimisant ainsi drastiquement les risques de péremption et le gaspillage.

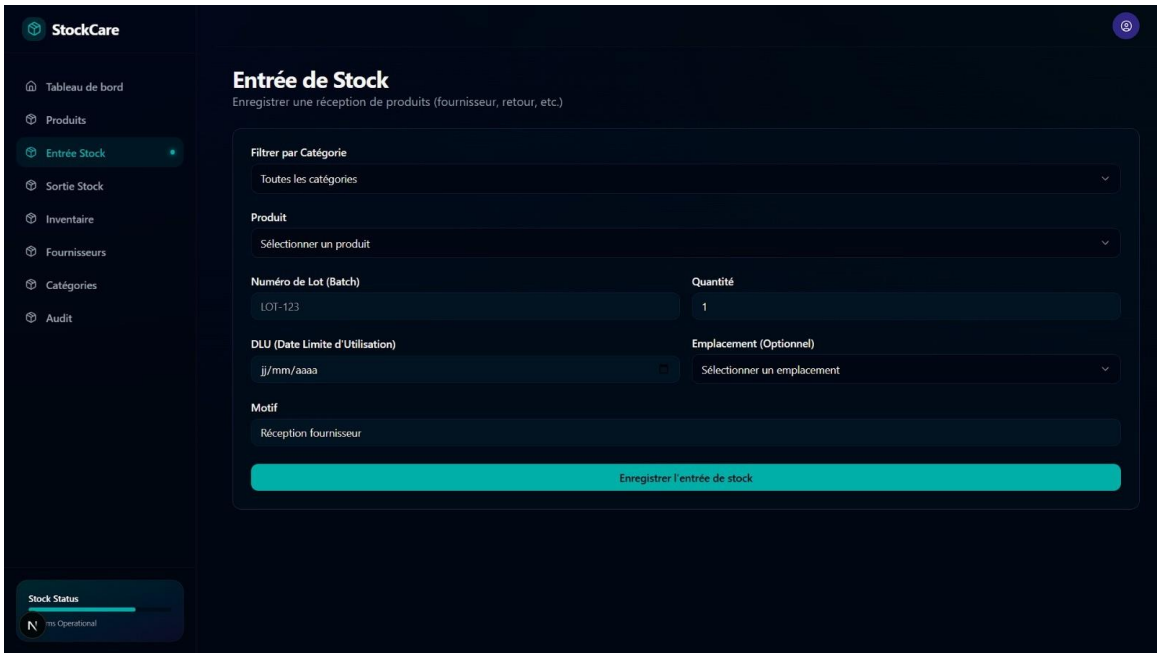


FIGURE 5.6 – Interface d'entrée de stock

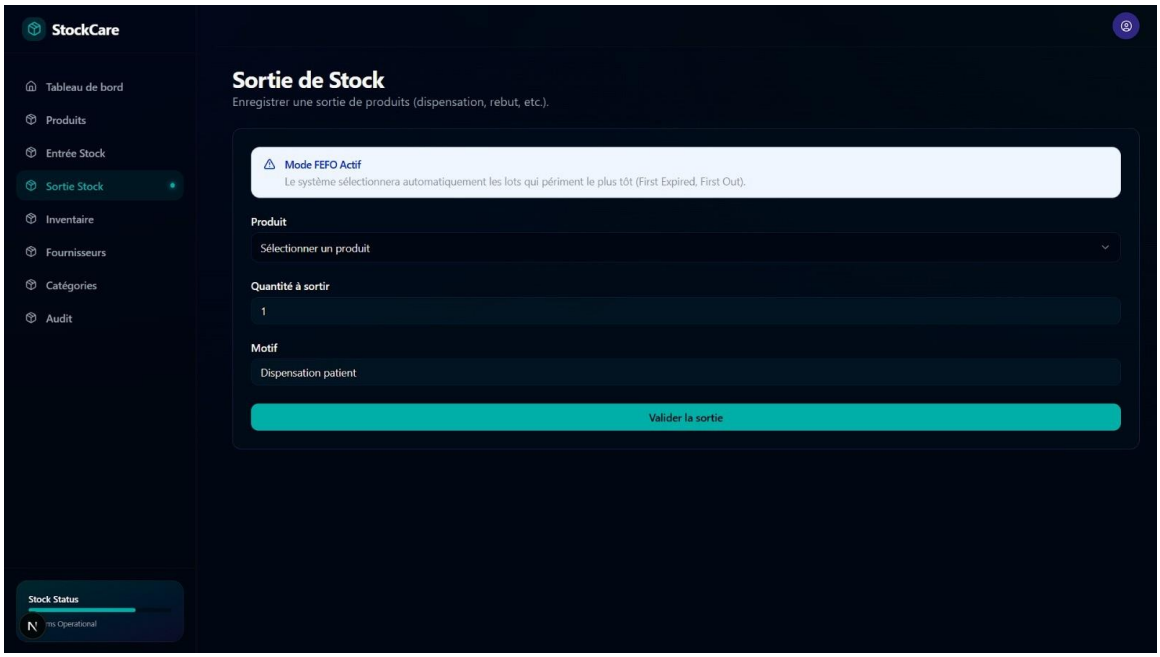


FIGURE 5.7 – Interface de sortie de stock

5.1.5 Inventaire et Anomalies

L'écran d'Inventaire facilite le comptage périodique en confrontant le stock théorique (calculé par le système) au stock réel (compté sur le terrain), permettant de régulariser les écarts rapidement.

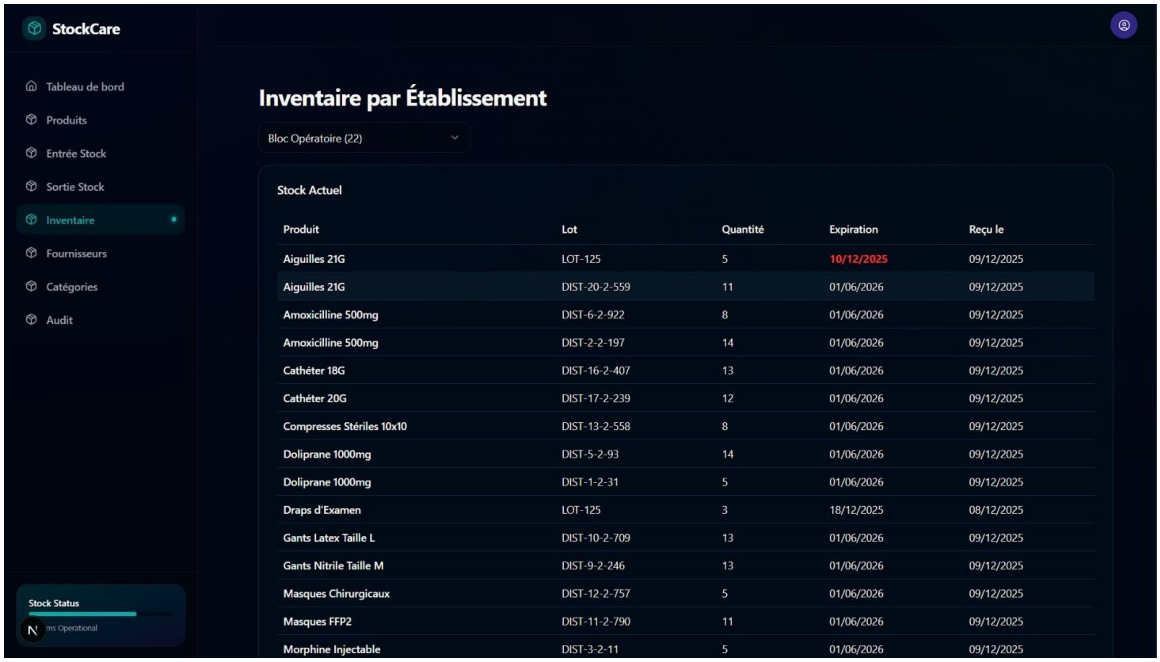


FIGURE 5.8 – Interface de l'inventaire

En parallèle, le module de Prédiction utilise l'historique des données de consommation pour anticiper les ruptures de stock et les sur-stocks à venir, transformant ainsi une gestion

purement réactive en un pilotage proactif des ressources.

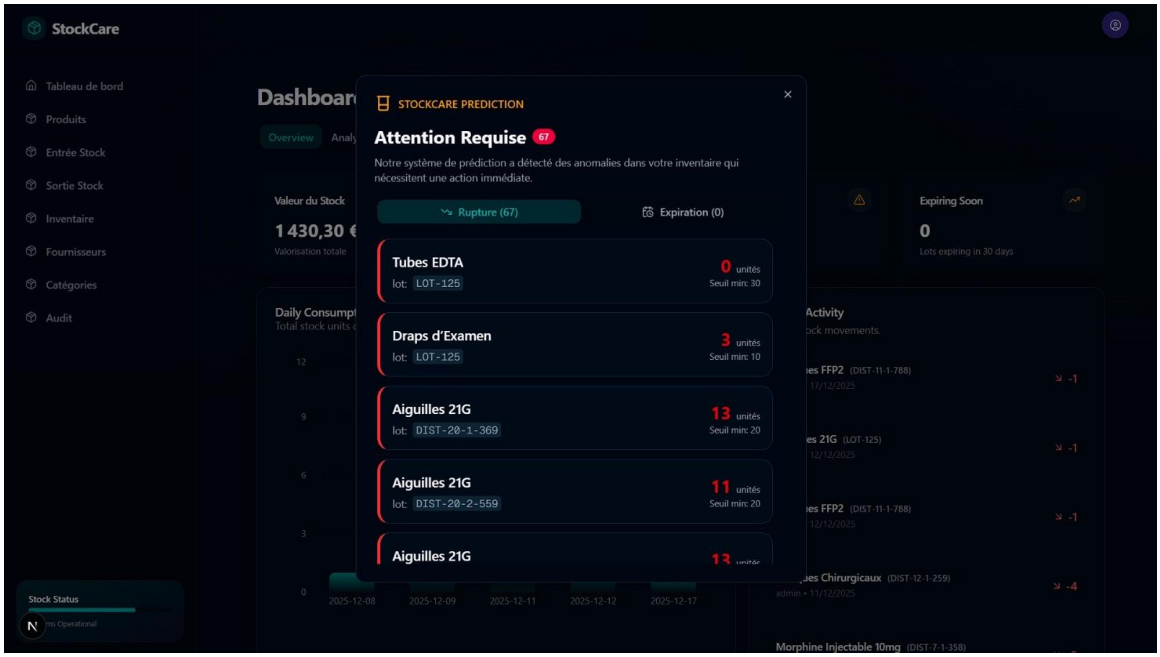


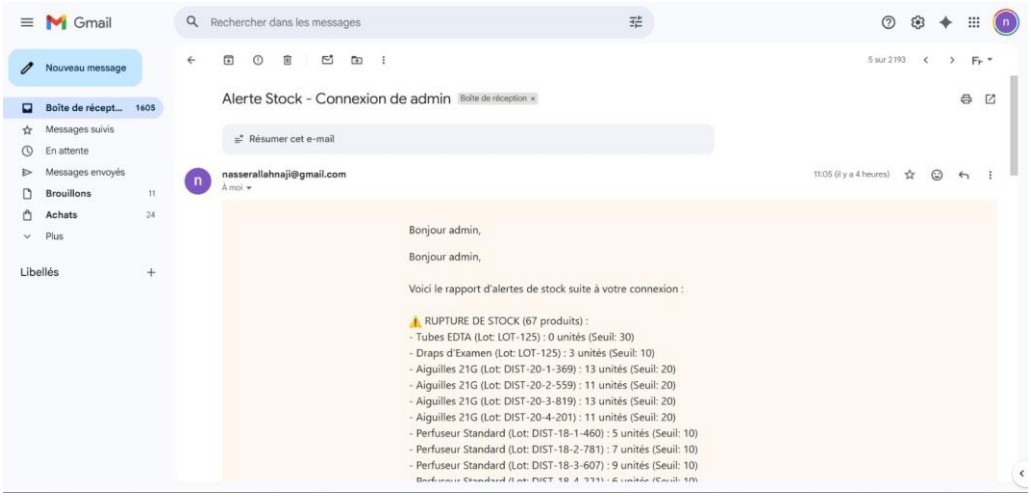
FIGURE 5.9 – Interface du système de prédiction des anomalies

5.1.6 Gestion des Stocks et Alertes

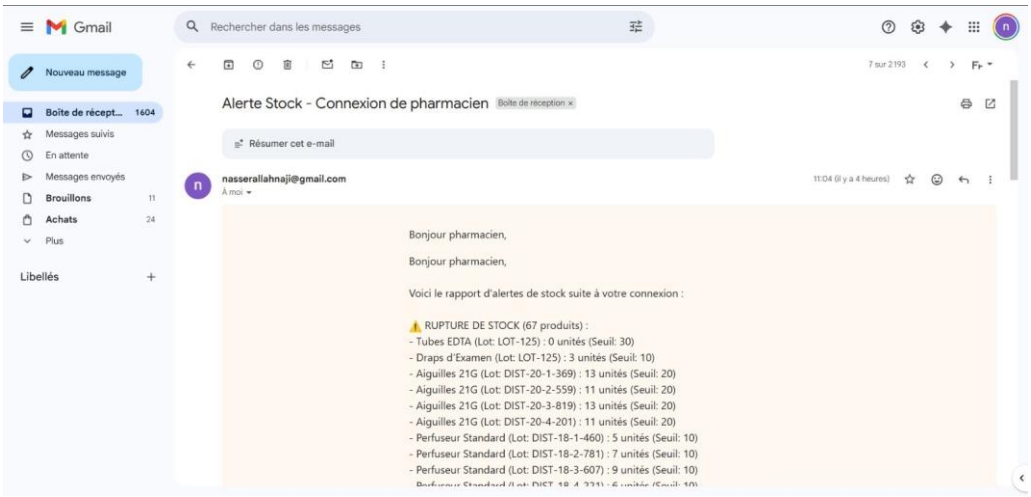
La sécurité des patients dépend de la réactivité des équipes logistiques. Ce module centralise toutes les notifications critiques. L'interface permet de visualiser les états de stock en temps réel, et le système d'alerte notifie automatiquement les acteurs concernés dès qu'un seuil critique est atteint ou qu'une date de péremption approche.

Flux de Notification Multi-Acteurs

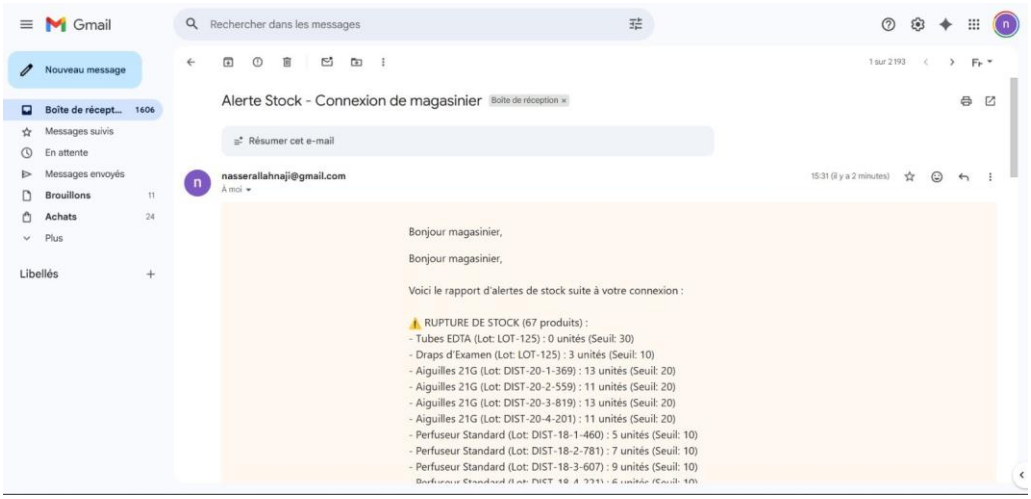
Les captures ci-dessous illustrent les alertes reçues par chaque acteur clé du système :



(a) Vue Admin : Tableau de pilotage des alertes



(b) Vue Pharmacien : Notification de péremption imminente



(c) Vue Magasinier : Ordre de réapprovisionnement

FIGURE 5.10 – Système centralisé de gestion des alertes StockCare

5.2 Tableau de Bord et KPI (Indicateurs Clés)

Un élément central de StockCare est son intelligence métier.

5.2.1 Indicateurs de Performance Suivis

1. Taux de Rupture (T_R) : Mesure la disponibilité des produits critiques.

$$T_R = \frac{\text{Demandes non satisfaites}}{\text{Total demandes}} \times 100 \tag{5.1}$$

2. Taux de Péremption (T_P) : Mesure les pertes financières.

$$T_P = \frac{\text{Valeur stock périmé}}{\text{Valeur stock total}} \times 100 \tag{5.2}$$

5.3 Structure de la Base de Données

L'implémentation repose sur un schéma relationnel robuste (PostgreSQL).

```sql
CREATE TABLE stock_items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    product_id INT NOT NULL,
    lot_number VARCHAR(100) NOT NULL,
    quantity INT NOT NULL CHECK (quantity >= 0),
    expiry_date DATE NOT NULL,
    UNIQUE (product_id, lot_number)
);
```

Listing 5.1 – Extrait de la table stock_items

Conclusion

La présentation détaillée des interfaces démontre que StockCare n'est plus un concept, mais une application fonctionnelle et complète. Nous avons réussi à traduire les besoins utilisateurs en écrans intuitifs et performants, respectant les objectifs de traçabilité et d'efficacité fixés au départ. Cette réalisation technique finalise notre démarche de conception et de développement présentée tout au long de ce rapport.

Conclusion Générale

Au terme de ce projet de fin d'année, nous éprouvons une grande satisfaction face au chemin parcouru. La réalisation de la plateforme **StockCare** nous a permis de relever un défi ambitieux : concevoir une solution complète répondant aux exigences complexes de la logistique hospitalière.

D'un point de vue **technique**, ce projet a été une formidable opportunité de montée en compétence. Nous avons maîtrisé une architecture Full Stack moderne (NestJS, Next.js), appréhendé les enjeux de la conteneurisation avec Docker, et exploré le monde de la Business Intelligence avec Power BI. La robustesse et la scalabilité de l'application témoignent de la qualité de notre apprentissage.

D'un point de vue **fonctionnel**, nous avons apporté une réponse concrète aux maux qui affligent la gestion de stock traditionnelle : le manque de traçabilité, les ruptures inattendues et le gaspillage coûteux. StockCare offre aujourd'hui un outil ergonomique et intelligent capable d'améliorer significativement le quotidien des pharmaciens et magasiniers.

Enfin, sur le plan **humain et organisationnel**, ce projet nous a enseigné l'importance de la méthodologie, du travail en équipe et de la communication. La gestion des délais, la répartition des tâches et l'adaptation aux contraintes ont été autant d'apprentissages précieux pour notre future carrière d'ingénieurs.

Cependant, StockCare n'est pas une fin en soi, mais un début. Nous envisageons déjà de futures évolutions, telles que l'intégration de l'Intelligence Artificielle pour affiner les prédictions de consommation, ou le développement d'une application mobile pour faciliter les inventaires sur le terrain. Ce projet restera pour nous une expérience fondatrice, marquant la transition entre notre formation académique et le monde professionnel.

Bibliographie

[1] NestJS Documentation, *Une architecture progressive pour Node.js*, https://docs.nestjs.com/.

[2] Next.js Documentation, *The React Framework for the Web*, https://nextjs.org/docs.

[3] Ken Schwaber et Jeff Sutherland, *Le Guide Scrum*, 2020.

[4] Pierre Médan, *Logistique et Supply Chain Management*, Dunod, 2008.

[5] Docker Inc., *Docker Documentation*, https://docs.docker.com/.

[6] PostgreSQL Global Development Group, *PostgreSQL 15 Documentation*, https://www.postgresql.org/docs/.

Annexe A

Annexes

A.1 Script de lancement Docker

```
git clone https://github.com/votre-repo/stockcare.git
cd stockcare
docker-compose up --build -d
```

Listing A.1 – Lancement rapide