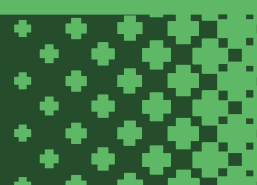


# **GSB-Stock**

## RAPPORT PROJET

Bonvallet Lucas Bts sio Slam- 2024





---

# Sommaire

- 
- 01. - Contexte
  - 02. – Expression besoins
  - 03. – Objectifs
  - 04. – Analyse fonctionnelle
  - 05. – Déploiement
  - 06. – Manuel utilisateur
  - 07. – Conclusion
-



---

# Contexte

## **Le domaine d'activité en question :**

Le secteur pharmaceutique représente une industrie hautement lucrative où les opérations de fusion-acquisition sont très actives. Les récentes consolidations entre laboratoires ont donné naissance à des entités gigantesques au sein desquelles l'organisation du travail reste souvent structurée selon les anciennes méthodes. Des controverses récentes autour de certains médicaments ou molécules ayant entraîné des complications médicales ont suscité des critiques concernant une partie de l'activité des laboratoires : la visite médicale, souvent considérée comme un terrain où des arrangements sont conclus entre l'industrie et les praticiens, et du moins un domaine d'influence opaque

## **L'entreprise :**

Le laboratoire Galaxy Swiss Bourdin (GSB) est le résultat de la fusion entre le géant américain Galaxy, spécialisé dans le domaine des maladies virales telles que le SIDA et les hépatites, et le conglomérat européen Swiss Bourdin, axé sur des médicaments plus traditionnels. Swiss Bourdin, à son tour, était déjà le fruit de l'union de trois petits laboratoires. En 2009, ces deux mastodontes de l'industrie pharmaceutique ont combiné leurs expertises pour former un leader dans ce domaine. Galaxy Swiss Bourdin Europe a choisi Paris comme base administrative, tandis que le siège social de la multinationale est implanté à Philadelphie, en Pennsylvanie, aux États-Unis. La France a été sélectionnée comme pilote pour l'amélioration du suivi des activités de visite.

## **Réorganisation :**

Une conséquence de cette fusion est la recherche d'une optimisation de l'activité du groupe ainsi formé, en réalisant des économies d'échelle dans la production et la distribution des médicaments. Cela implique une restructuration nécessaire et une vague de licenciements, tout en capitalisant sur les points forts des deux laboratoires concernant les produits concurrents. Actuellement, l'entreprise compte 480 visiteurs médicaux en France métropolitaine, y compris la Corse, et 60 dans les départements et territoires d'outre-mer (DTOM). Les territoires sont répartis en 6 secteurs géographiques : Paris-Centre, Sud, Nord, Ouest, Est, ainsi que DTOM Caraïbes-Amériques et DTOM Asie-Afrique.



---

## **Expression des besoins**

---

### **Gestion des utilisateurs :**

- Permettre l'ajout, la modification et la consultation des utilisateurs avec des rôles appropriés (administrateur, utilisateur).
- Créer une base de données d'utilisateurs pour gérer les accès au système.

### **Gestion des stocks :** Permettre l'ajout, la modification et la consultation des stocks de manière conviviale et intuitive.

- Mettre en place une base de données pour stocker les informations relatives aux médicaments, équipements et autres ressources.

### **Gestion des commandes :** Assurer le contrôle des entrées et sorties de stocks lors du traitement des commandes.

- Faciliter l'ajout et la consultation des commandes de manière conviviale et intuitive.
- Établir une base de données pour stocker les informations relatives aux commandes.

### **Sécurité / Confidentialité :**

- Mettre en place des mécanismes d'authentification des utilisateurs pour assurer un accès sécurisé au système.
  - Chiffrer les données sensibles, notamment les mots de passe des utilisateurs.
  - Garantir la confidentialité des données en mettant en place des mesures de protection adéquates.
-



---

# Les Objectifs

Le projet vise à permettre au laboratoire pharmaceutique GSB de gérer efficacement ses stocks de médicaments, équipements et autres ressources. Le logiciel, nommé GSB-STOCK comportera une base de données pour les stocks et les commandes, facilitant ainsi le suivi des entrées et sorties du stock. De plus, une base d'utilisateurs sera mise en place pour gérer les accès au système, avec une validation nécessaire par un administrateur.

Pour atteindre ces objectifs, GSB-STOCK devra offrir une interface conviviale et intuitive permettant d'ajouter, de modifier et de consulter les commandes et les stocks. Une attention particulière sera portée à la sécurité et à la confidentialité des données. Des mesures de protection appropriées seront mises en œuvre, telles que l'authentification des utilisateurs et le chiffrement des données sensibles comme les mots de passe.



# Analyse fonctionnelle

Langages et technologies utilisés

PHP

SQL

CSS

XAMPP (X, Apache (serveur Web), MariaDB (système de gestion de bases de données relationnelles), PHP, PERL)

Git, GitHub

Visual Studio Code

## Liens GitHub

- <https://github.com/Lulubo38/GSBstok>

## Présentation des technologies utilisées :


PHP (PHP: Hypertext Preprocessor) est un langage de programmation côté serveur populaire et largement utilisé pour le développement web. Il est principalement utilisé pour générer des pages web dynamiques et interagir avec des bases de données. PHP offre une grande flexibilité et une large gamme de fonctionnalités pour créer des applications web puissantes.

SQL (Structured Query Language) est un langage de programmation spécialement conçu pour gérer les bases de données relationnelles. Il permet de créer, modifier et interroger des bases de données pour stocker, récupérer et manipuler des informations. SQL est essentiel dans le développement de systèmes de gestion de bases de données et joue un rôle clé dans la manipulation et la gestion des données.

CSS (Cascading Style Sheets) est un langage de feuille de style utilisé pour décrire la présentation et l'apparence des documents HTML. Il permet de contrôler les aspects visuels d'une page web, tels que les couleurs, les polices, les marges, les mises en page, etc. CSS offre une séparation claire entre la structure HTML et le style, permettant ainsi une conception web plus modulaire et maintenable.

XAMPP (X, Apache, MariaDB, PHP, PERL) est un ensemble de logiciels open source qui fournit un environnement de développement web complet. En utilisant XAMPP, les développeurs peuvent configurer un environnement de développement local pour tester et exécuter leurs applications web avant de les déployer sur un serveur en production.

Git est un système de contrôle de version distribué qui permet aux développeurs de gérer et de suivre les modifications de leur code source. Il permet de créer des branches, de fusionner des modifications et de revenir à des versions antérieures du code. Git facilite la collaboration et la gestion des projets de développement de logiciels



GitHub est une plateforme de développement logiciel basée sur Git, où les développeurs peuvent héberger et partager leurs projets Git de manière centralisée. Il fournit également des fonctionnalités de suivi des problèmes, de gestion des demandes de fusion et de collaboration entre développeurs. GitHub est largement utilisé dans la communauté du développement logiciel open source et facilite le partage et la contribution aux projets.

Visual Studio Code (VS Code) est un éditeur de code source léger et très populaire développé par Microsoft. Il prend en charge de nombreux langages de programmation et offre une variété de fonctionnalités telles que la coloration syntaxique, l'autocomplétion, le débogage, la gestion des extensions et l'intégration avec des outils de contrôle de version. VS Code est apprécié pour sa facilité d'utilisation, sa personnalisation et sa capacité à s'adapter à différentes technologies de développement.

Liaison avec la base de données :

Dans mon application, la connexion avec la base de données se fait à l'aide de PHP Data Object (PDO). PDO est une extension de PHP qui fournit une interface cohérente pour accéder aux bases de données. L'avantage de son utilisation est que nous pouvons travailler avec plusieurs systèmes de base de données différents sans changer la logique PHP. PDO permet également la création de requêtes préparées afin d'éviter les injections SQL. Voici le code correspondant à ma classe Database :

```
espace Membre > database.php
1  <?php
2  $host = 'localhost'; // Hôte de la base de données
3  $dbname = 'gsb'; // Nom de la base de données
4  $username = 'root'; // Nom d'utilisateur de la base de données
5  $password = ''; // Mot de passe de la base de données
6
7  try {
8      $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
9      // Définir le mode d'erreur de PDO à exception
10     $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
11 } catch(PDOException $e) {
12     die("Impossible de se connecter à la base de données : " . $e->getMessage());
13 }
14
15
```

## Présentation page de connexion / inscription

Bienvenue sur notre plateforme de gestion des stocks ! Notre page de connexion, index.php, constitue le point d'entrée sécurisé de notre système, permettant aux utilisateurs autorisés d'accéder à leurs comptes en toute simplicité.

Vue d'ensemble :

La page de connexion présente une interface épurée et fonctionnelle, conçue pour faciliter l'accès rapide aux comptes utilisateurs. Son design intuitif garantit une expérience de connexion fluide et sécurisée.

Authentification Sécurisée :

À leur arrivée sur index.php, les utilisateurs sont invités à saisir leurs identifiants de connexion dans les champs dédiés. Notre système d'authentification sécurisé vérifie les informations fournies, garantissant l'accès exclusivement aux utilisateurs autorisés.

Les utilisateurs seront redirigés en fonction de leur rôles (admin, user)

```
// Rechercher l'utilisateur dans la table `utilisateurs` avec son rôle
$recupUser = $pdo->prepare('SELECT u.id_utilisateur, u.nom, u.mot_de_passe, r.id_role FROM utilisateurs u INNER JOIN roles r ON u.id_role = r.id_role WHERE u.nom = :nom');
$recupUser->execute(array($nom));
$user = $recupUser->fetch();

// Si l'utilisateur est trouvé dans la table `utilisateurs`, vérifier le mot de passe
if ($user && password_verify($mot_de_passe, $user['mot_de_passe'])) {
    $_SESSION['nom'] = $nom;
    $_SESSION['id'] = $user['id_utilisateur'];
    $_SESSION['id_role'] = $user['id_role']; // Utiliser le rôle de l'utilisateur

    // Redirection en fonction du rôle de l'utilisateur
    if ($user['id_role'] == 1) {
        // L'utilisateur est un administrateur
        header('location: dashboard_admin.php');
        exit();
    } elseif ($user['id_role'] == 2) {
        // L'utilisateur est un utilisateur
        header('location: dashboard.php');
        exit();
    }
}
```



Si l'utilisateur n'est pas déjà inscrit il peut accéder a la page d'inscription

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (!empty($_POST['nom']) && !empty($_POST['email']) && !empty($_POST['mot_de_passe'])) {
        $nom = htmlspecialchars($_POST['nom']);
        $email = htmlspecialchars($_POST['email']);
        $mot_de_passe = password_hash($_POST['mot_de_passe'], PASSWORD_DEFAULT); // Hash du mot de passe

        // Vérification si l'utilisateur existe déjà
        $checkUser = $pdo->prepare('SELECT id_utilisateur FROM utilisateurs WHERE nom = ?');
        $checkUser->execute(array($nom));
        $existingUser = $checkUser->fetch();

        if ($existingUser) {
            echo "Cet utilisateur existe déjà.";
        } else {
            // Préparation de la requête d'insertion dans la table utilisateurs
            $insertUser = $pdo->prepare('INSERT INTO utilisateurs (nom, email, mot_de_passe) VALUES (?, ?, ?)');
            $insertUser->execute(array($nom, $email, $mot_de_passe));

            // Sélection de l'utilisateur nouvellement inscrit
            $recupUser = $pdo->prepare('SELECT id_utilisateur FROM utilisateurs WHERE nom = ?');
            $recupUser->execute(array($nom));
            $user = $recupUser->fetch();
        }
    }
}
```

Si l'utilisateur parviens à se connecter en tant qu'utilisateur alors il peut accéder au Dashboard ou il Pourrat Retrouver le stock dans une liste déroulante. Le stock et séparer en 2 catégories Médicament et matériel avec la quantité qui leur est associer.

```
// Vérifier si l'utilisateur est connecté
if (!isset($_SESSION['id'])) {
    // Rediriger vers la page de connexion
    header("Location: index.php");
    exit;
}

// Récupérer les stocks depuis la base de données
$requete_stocks = $pdo->query('SELECT id_stock, nom, quantite_disponible, type FROM stocks');
$stocks = $requete_stocks->fetchAll();
?>

<!DOCTYPE html>
<html lang="fr">
<head>
    <title>Consultation des Stocks</title>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
```

L'utilisateur peut accéder à la page lui permettant de passer commande qui devra être validée par un admin avant d'être effectuée.

```
<?php
// Inclusion du fichier de connexion à la base de données
require_once 'database.php';

// Récupérer les stocks disponibles
$requete_stocks = $pdo->query('SELECT id_stock, nom, quantite_disponible FROM stocks');
$stocks = $requete_stocks->fetchAll();

foreach ($stocks as $stock) {
    echo '<option value="' . $stock['id_stock'] . '"> . $stock['nom'] . ' (Quantité disponible : ' . $stock['quantite_disp
}
?>
```

L'utilisateur peut retourner à l'accueil via le bouton retour et il peut également se déconnecter grâce à log out

```
espace Membre > 🚪 logout.php
1  <?php
2  session_start();
3  session_destroy();
4  header("Location: index.php");
5  exit();
6  ?>
```

Si l'utilisateur qui se connecte et détecte comme étant un admin il accèdera à dashboard\_admin même interface que pour l'utilisateur mais avec 2 boutons supplémentaires disponibles la possibilité de valider une commande créée par les utilisateurs

```
// Récupérer l'ID de la demande de commande depuis l'URL
$id_demande = $_GET['id'];

// Mettre à jour le statut de la demande de commande dans la base de données
$requete_update_statut = $pdo->prepare('UPDATE demandes_commandes SET statut = "validée" WHERE id_demande = ?');
$requete_update_statut->execute([$id_demande]);

// Récupérer la quantité demandée et l'ID du stock correspondant à la demande
$requete_info_demande = $pdo->prepare('SELECT id_stock, quantite_demandee FROM demandes_commandes WHERE id_demande = ?');
$requete_info_demande->execute([$id_demande]);
$info_demande = $requete_info_demande->fetch();

$id_stock = $info_demande['id_stock'];
$quantite_demandee = $info_demande['quantite_demandee'];

// Mettre à jour la quantité disponible dans le stock correspondant
$requete_update_stock = $pdo->prepare('UPDATE stocks SET quantite_disponible = quantite_disponible - ? WHERE id_stock = ?');
$requete_update_stock->execute([$quantite_demandee, $id_stock]);

// Rediriger vers la page dashboard.php
header("Location: dashboard_admin.php");
exit;
```

L'admin peut également modifier le stock (modification quantité, description, ajout nouveau stock)

```
// Traitement des actions de l'administrateur
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST['ajouter_stock'])) {
        // Récupérer les données du formulaire d'ajout de stock
        $nom_nouveau_stock = $_POST['nom_nouveau_stock'];
        $quantite_nouveau_stock = $_POST['quantite_nouveau_stock'];
        $description_nouveau_stock = $_POST['description_nouveau_stock'];
        $type_nouveau_stock = $_POST['type_nouveau_stock'];

        // Insérer le nouveau stock dans la base de données
        $requete_insert_stock = $pdo->prepare('INSERT INTO stocks (nom, quantite_disponible, description, type) VALUES (?, ?, ?, ?)');
        $requete_insert_stock->execute([$nom_nouveau_stock, $quantite_nouveau_stock, $description_nouveau_stock, $type_nouveau_stock]);

        // Enregistrer le mouvement correspondant dans la table des mouvements
        $id_stock = $pdo->lastInsertId(); // Obtenez l'ID du nouveau stock inséré
        $type_mouvement = 'entree'; // Type de mouvement pour une nouvelle entrée de stock
        $quantite_mouvement = $quantite_nouveau_stock; // La quantité ajoutée est égale à la quantité du nouveau stock
        $date_mouvement = date("Y-m-d H:i:s"); // Date et heure actuelles

        $requete_insert_mouvement = $pdo->prepare('INSERT INTO mouvements (id_stock, type_mouvement, quantite, date_mouvement) VALUES (?, ?, ?, ?)');
        $requete_insert_mouvement->execute([$id_stock, $type_mouvement, $quantite_mouvement, $date_mouvement]);
    } elseif (isset($_POST['supprimer_stock'])) {
        // Récupérer l'ID du stock à supprimer
    }
}
```

Je vous laisse 2 identifiants afin de vous faciliter la connexion

Admin = adm mdp= lulu

User = 1 mdp= 1

Chaque mouvement de stock et répertorié dans la base de données grâce à la table mouvements

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	id_mouvement	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier  Supprimer  Plus
<input type="checkbox"/> 2	id_stock	int(11)			Oui	NULL			Modifier  Supprimer  Plus
<input type="checkbox"/> 3	type_mouvement	enum('entree', 'sortie')	utf8mb4_general_ci		Oui	NULL			Modifier  Supprimer  Plus
<input type="checkbox"/> 4	quantite	int(11)			Oui	NULL			Modifier  Supprimer  Plus
<input type="checkbox"/> 5	date_mouvement	datetime			Oui	current_timestamp()			Modifier  Supprimer  Plus
<input type="checkbox"/> 6	id_commande	int(11)			Oui	NULL			Modifier  Supprimer  Plus



# Déploiement

Afin de finaliser ce projet et le préparer pour la production, j'ai choisi de le déployer en ligne sur un serveur web avec un nom de domaine. Voici les étapes que j'ai entreprises :

Le serveur que j'utilise est un serveur Linux Ubuntu hébergé. Il s'agit d'un serveur de type "VPS" (Virtual Private Server) que je loue auprès d'un hébergeur. Ce type de service VPS est pratique car il fournit un serveur Ubuntu (ou autre) prêt à l'emploi, avec une adresse IP publique et une connexion SSH.

Pour commencer, j'ai installé Apache2, qui me servira à mettre en place le serveur web. Voici comment j'ai procédé :

```
Sudo apt install apache2 -y
```

Commandes pour démarrer le service et pour qu'il démarre également au démarrage du serveur :

```
Sudo systemctl start apache2  
Sudo systemctl enable apache2
```

Ensuite, pour la base de données, j'ai choisi MariaDB. MariaDB est un système de gestion de base de données relationnelle basé sur SQL. Voici les commandes que j'ai utilisées pour installer le service :

```
Sudo apt install mariadb-server -y
```

Pour sécuriser la base de données au minimum, j'ai exécuté la commande suivante :

```
Sudo mysql_secure_installation
```

Pour permettre au serveur web de fonctionner correctement, j'ai installé PHP avec les extensions pour Apache et MySQL :


```
Sudo apt install php libapache2-mod-php php-mysql -y
```

```
Sudo systemctl restart apache2
```

En ce qui concerne la configuration de MariaDB, voici les étapes que j'ai suivies :

```
CREATE DATABASE GSBstock ;
```

```
GRANT ALL PRIVILEGES ON GSBstock. * TO 'gsbstock'@'localhost' IDENTIFIED BY  
    'motdepasse' ;  
FLUSH PRIVILEGES ;
```



Enfin, pour configurer PHP, j'ai modifié le fichier php.ini pour définir la valeur de la directive session. auto\_start à 1 :

```
session. auto_start = 1
```

Remplissage de la base :

```
source /chemin/vers/le/script.sql;
```

Pour configurer apache j'ai créé un nouveau fichier de configuration gsbstock.conf à l'adresse :  
etc/apache2/sites-available/

## Importer le code source de l'application

Pour importer le code source de l'application, j'utilise Git pour cloner mon dépôt distant. Git est déjà installé par défaut sur mon serveur. J'utilise un dossier temporaire pour intégrer mon code afin d'éviter d'appliquer des modifications en production. Voici la commande pour cloner le dépôt :

```
git clone https://github.com/Lulubo38/GSBstok.git
```

Après avoir cloné le dépôt distant, je me positionne sur ma branche nommée "pull", qui contient une version minimale du code prête pour la production. La seule modification requise concerne les identifiants de connexion à la base de données, qui peuvent différer de ceux sur le dépôt distant. Voici la commande pour passer à la branche "pull" :

```
git checkout pull
```

Ensuite, je transfère ce dossier à la racine du serveur web.

Après avoir redémarré le serveur pour garantir la prise en compte de toutes les informations, le serveur web est déployé. Seule la configuration des entrées A au niveau DNS doit être effectuée auprès du fournisseur.



---

# Manuel Utilisateur

L'application est accessible via cet adresse 149.202.53.198

Voici des identifiants de connexions :

Administrateur : adm mdp :lulu

Utilisateur :1 mdp :1

Lors de la connexion, l'utilisateur est dirigé vers le "Dashboard", où il trouvera diverses informations importantes. Tous les éléments de ce tableau de bord sont interactifs, ce qui permet un accès rapide aux informations en un simple clic.

Tableau de Bord Admin

[Déconnexion](#) [Historique des commandes](#) [Gérer les stocks](#) [Commandes par mois](#)

Gestion du Stock

Afficher : 

Tous

smecta - Quantité : 30

brosse a dent - Quantité : 10

brique - Quantité : 4

Paracétamol - Quantité : 1



L'utilisateur peut voir le stock avec le nombre de produit restant et classer en fonction de s'il recherche un médicament ou du matériel.

Déconnexion

Historique des commandes

Gérer les stocks

Commandes par mois

Tableau de Bord Admin

Gestion du Stock

Afficher : Tous

smecta - Médicaments

brosse a dent - Matériel

brosse a dent - Quantité : 10

brique - Quantité : 4

Paracétamol - Quantité : 1

L'administrateur lui a accès a d'autre option il peut valider les commandes passer par les utilisateurs ce qui actualise le nombre restant dans le stock il peut également les refuser.

Retour au tableau de bord

ID Demande	ID Stock	Quantité demandée	Date Demande	Statut	Actions
12	5	1	2024-03-23 21:11:30	validee	<a href="#">Valider</a> <a href="#">Refuser</a>
15	1	15	2024-03-25 22:20:40	validee	<a href="#">Valider</a> <a href="#">Refuser</a>
16	4	1	2024-03-25 22:22:59	en_attente	<a href="#">Valider</a> <a href="#">Refuser</a>
17	1	2	2024-03-25 22:22:52	en_attente	<a href="#">Valider</a> <a href="#">Refuser</a>
21	1	1	2024-03-25 22:58:30	en_attente	<a href="#">Valider</a> <a href="#">Refuser</a>
22	1	20	2024-03-25 22:59:00	validee	<a href="#">Valider</a> <a href="#">Refuser</a>
23	1	3	2024-03-26 15:08:20	validee	<a href="#">Valider</a> <a href="#">Refuser</a>

L'administrateur peut également crée de nouveaux stocks ou modifier les existants, mais également les quantité

### Gérer les Stocks

[Retour au tableau de bord](#) [Déconnexion](#)

#### Ajouter un Nouveau Stock

Nom :  Quantité :  Description :  Type : Médicament

#### Liste des Stocks Existants

ID Stock	Nom	Quantité Disponible	Description	Type	Action
1	smecta	30	pour les maux de ventre	medicament	<input type="text"/> <input type="button" value="Modifier Quantité"/> <input type="button" value="Supprimer"/>
2	brosse a dent 10		souple	materiel	<input type="text"/> <input type="button" value="Modifier Quantité"/> <input type="button" value="Supprimer"/>
4	brique	4	yftydytd	materiel	<input type="text"/> <input type="button" value="Modifier Quantité"/> <input type="button" value="Supprimer"/>
5	Paracétamol	1	Antalgique • Antipyrétique	medicament	<input type="text"/> <input type="button" value="Modifier Quantité"/> <input type="button" value="Supprimer"/>

Il a également la possibilité de voir le nombre de commande qui a été effectuer par mois .

### Passation de Commandes

Produit :

smecta (Quantité disponible : 30)

Quantité :

Passer Commande

Voici la page qui permet à l'utilisateur de passer une commande.





# Conclusion :

En conclusion, le développement de ce projet PHP a été une expérience passionnante et enrichissante. La création de cette plateforme de gestion des stocks a permis de mettre en pratique divers concepts de programmation web tout en répondant à un besoin concret dans le domaine de la gestion d'inventaire.

Cependant, bien que le projet soit déjà fonctionnel, il reste encore de nombreuses fonctionnalités à ajouter pour le perfectionner davantage. Des améliorations telles que la gestion avancée des utilisateurs, la génération de rapports détaillés, ou encore l'intégration de notifications en temps réel pourraient grandement enrichir l'expérience des utilisateurs et rendre le système plus efficace.

De plus, l'aspect visuel de la plateforme est également un point à améliorer. L'ajout d'une interface utilisateur plus intuitive et attrayante contribuerait à rendre le site plus agréable à utiliser. À cet égard, la recherche d'une DA (Directrice Artistique) qualifiée pourrait être une étape clé pour donner une identité visuelle unique et professionnelle à notre application.

En somme, bien que le projet PHP actuel représente un jalon important dans le développement de notre solution de gestion des stocks, il reste encore un chemin à parcourir pour atteindre son plein potentiel. Avec un engagement continu et une vision orientée vers l'amélioration, nous sommes confiants que notre plateforme évoluera pour devenir une référence dans le domaine de la gestion d'inventaire.