Rapport de Mise à Jour et d'Intégration des Améliorations

Ce document récapitule les récentes modifications techniques et fonctionnelles apportées à l'application web MyBrickStore.

1. Avant-propos : Démarche Qualité et Documentation

Un travail a été réalisé sur la qualité logicielle.

Dans une optique de professionnalisation, nous avons mené une phase de recherche sur les **standards de documentation industriels** pour chaque langage. L'objectif était de dépasser le simple commentaire de code pour atteindre une documentation normative :

- **Méthodologie :** Adoption stricte des balises standard (`@param`, `@return`, `@throws`, `@var`) après étude des conventions (Oracle pour Java, PSR pour PHP).
- **Rigueur :** 100% des fichiers (Classes, Contrôleurs, Headers C, Scripts SQL) ont été documentés.
- **Résultat :** Cette rigueur a permis la **génération automatique** de documentations techniques consultables sous forme de sites web (HTML) via les outils de référence : **Javadoc** (Java), **PHPDoc** (PHP) **Doxygen** (C) et **DBDocs** (SQL).

Consulter la documentation technique : https://alkzhab.github.io/MyBrickStore-Doc/

2. Module C (Algorithmique et Traitement)

Les améliorations en C se concentrent sur l'optimisation des algorithmes de génération de mosaïques pour améliorer la pertinence des résultats et la rentabilité.

Refonte de l'algorithme "Forme Libre"

- **Amélioration :** Changement complet de la logique pour prioriser l'utilisation de grandes briques.
- **Intégration :** L'algorithme a été réécrit pour chercher d'abord à placer les plus grandes pièces possibles. Cela réduit le nombre total de briques nécessaires et le coût final. Cette modification permet désormais de distinguer nettement ce mode du mode "Minimisation", qui produisait auparavant des résultats trop similaires.

Ajustement de l'algorithme "Rentabilité"

- **Amélioration :** Modification légère pour équilibrer qualité et coût.
- **Intégration :** Ajustement des poids dans la fonction de calcul pour favoriser une meilleure qualité visuelle tout en conservant l'objectif de réduction des coûts.

3. Module Java (Logique Métier et Administration)

Le module Java a été enrichi pour lier l'interface d'administration aux fonctionnalités backend et assurer la cohérence des données financières.

Liaison des Commandes Admin

- **Amélioration :** La page Administrateur est désormais fonctionnelle et connectée au backend.
- **Intégration :** Implémentation des méthodes permettant de déclencher les actions de minage, de commande proactive et de commande spécifique directement depuis l'interface utilisateur.

Calcul et Stockage des Prix Réels

- **Amélioration :** Remplacement des prix théoriques par des prix réels basés sur des commandes simulées, rendant les mosaïques plus abordables.
- **Intégration :** Création d'un script Java dédié qui effectue une commande de 10 unités pour chaque pièce, divise le coût total par 10 pour obtenir un prix unitaire précis, et met à jour la base de données avec ces valeurs.

4. Module SQL (Base de Données)

Les interventions SQL visent à garantir l'intégrité des données critiques et à faciliter les calculs de gestion des stocks.

Calcul du Seuil Critique

- **Amélioration :** Automatisation de la surveillance des stocks.
- **Intégration :** Création d'une **Vue (View)** SQL dédiée qui calcule dynamiquement le seuil critique des stocks.

Mise à jour des Prix

- **Amélioration :** Persistance des données financières.
- **Intégration :** Mise à jour des tables pour inclure les prix réels des briques calculés par le module Java.

Immutabilité des Commandes (Sécurité)

- **Amélioration :** Garantie que les commandes passées ne peuvent pas être modifiées a posteriori.
- **Intégration :** Développement et ajout de 6 triggers sur les tables factoryOrder, factoryBrick et factoryOrderDetail pour bloquer toute tentative de modification après insertion.

5. Module PHP (Interface Web et Paiement)

L'accent a été mis sur l'expérience utilisateur (UX) et la mise en place du système de paiement.

Amélioration de l'Ergonomie des Formulaires (UX/UI) :

- **Visibilité du Mot de Passe :** Ajout d'une fonctionnalité interactive (icône "Œil") en JavaScript permettant à l'utilisateur de révéler ou masquer son mot de passe lors de la saisie, réduisant ainsi les erreurs de frappe.
- **Double Validation :** Renforcement de la sécurité à l'inscription via l'ajout d'un champ obligatoire **"Confirmer le mot de passe"**, assurant la cohérence des identifiants saisis.

Sécurisation du Routing (Expérience Utilisateur)

- **Amélioration :** Protection des données utilisateurs
- **Intégration :** Le système de routage impose désormais une authentification stricte pour accéder aux outils de création (Upload d'image). Ce choix technique garantit la persistance des projets utilisateurs et évite la perte de données (paniers abandonnés) liée aux sessions "invités".

Système de Paiement

- **Amélioration :** Possibilité de tester les transactions financières.
- **Intégration :** Intégration de l'API PayPal Sandbox pour simuler les paiements en environnement de développement.