Projet de Session : CryptoTrade Plateforme d'Échange de Crypto-Monnaies (Simulation)

Enseignant : Esaie Kuitche Kamela, Ph.D

8 février 2025

1 Introduction

Dans ce projet, vous allez concevoir et développer une plateforme de simulation d'échange de crypto-monnaies en PHP avec une base de données MySQL. Le but est d'appliquer les concepts de programmation orientée objet (POO), manipulation de bases de données, AJAX et gestion des sessions.

2 Objectifs pédagogiques

- Appliquer la **programmation orientée objet (POO)** en PHP.
- Manipuler une **base de données MySQL** efficacement.
- Implémenter des fonctionnalités **AJAX** pour des mises à jour en temps réel.
- Gérer **les sessions et l'authentification** des utilisateurs.
- Implémenter une **simulation dynamique réaliste** de l'évolution des crypto-monnaies.
- Créer une **interface utilisateur interactive** pour gérer les transactions.
- Intégrer un **système de paiement avec Stripe en mode test**.
- Implémenter des **fonctions mathématiques avancées** pour la gestion et l'analyse des crypto-monnaies.
- Ajouter un **système d'alerte** pour informer les utilisateurs de variations importantes de leurs cryptos.
- Limiter le **nombre de transactions par jour**, configurable par l'admin.
- Protéger la plateforme contre les **attaques par injection SQL**.

— Mettre en place un **système de logs** permettant l'audit des actions des utilisateurs.
— Exploiter **les notions vues en cours** : superglobales, variables de session, fonctions, etc.
— Ajouter une **authentification à deux facteurs (2FA)** pour renforcer la sécurité.
— Implémenter un **système d'invitation avec jeton d'inscription** pour restreindre l'accès aux utilisateurs autorisés.

3 Technologies utilisées

— **Backend** : PHP (POO), MySQL
— **Frontend** : HTML, CSS, JavaScript (AJAX pour les requêtes dynamiques)
— **Frameworks/Libraries** : Bootstrap, jQuery (facultatif), Chart.js (pour les graphiques)
— **Architecture** : Modèle MVC
— **Authentification** : Sessions PHP, 2FA (Google Authenticator ou Email OTP)
— **Paiement** : Stripe (mode test)

4 Fonctionnalités demandées

1. Gestion des utilisateurs et administrateurs

— Création d'un **super administrateur** capable de gérer la plateforme.
— Inscription et connexion des **utilisateurs** avec validation via un **jeton de souscription** (fourni par l'admin).
— Gestion des comptes utilisateurs : activation, désactivation, suppression.
— Authentification sécurisée avec **protection contre les injections SQL**.
— Implémentation d'une **authentification à deux facteurs (2FA)** via **Google Authenticator** ou **OTP par email**.
— Implémentation d'un **système d'invitation** : inscription possible uniquement avec un **jeton d'invitation généré par l'admin**.

2. Gestion des crypto-monnaies

- Le super admin peut ajouter de nouvelles crypto-monnaies à la plateforme (nom, symbole, prix initial).
- Stockage des informations sur chaque crypto-monnaie dans la base de données.
- Consultation de la liste des crypto-monnaies disponibles par tous les utilisateurs.

3. Simulation du marché des crypto-monnaies

- Implémentation d'un **script PHP qui met à jour le prix des crypto-monnaies** toutes les secondes.
- **Les variations doivent être réalistes** :
  - Petite volatilité : ±0,1% à ±0,5% par seconde pour les cryptos stables.
  - Volatilité moyenne : ±0,5% à ±2% par seconde pour les cryptos courantes.
  - Forte volatilité : ±2% à ±5% par seconde pour les cryptos très volatiles.
  - Implémentation d'un **modèle mathématique basé sur les moyennes mobiles et l'historique des variations**.
- Stockage des prix mis à jour dans une table dédiée pour permettre l'analyse des tendances.

4. Achat et vente de crypto-monnaies

- Les utilisateurs connectés peuvent acheter et vendre des crypto-monnaies avec leur solde disponible.
- Un historique des transactions doit être conservé en base de données.
- Mise à jour en temps réel du portefeuille utilisateur après chaque transaction.
- **Paiement via Stripe en mode test** pour acheter des jetons utilisables sur la plateforme.
- **Limitation du nombre de transactions par jour**, configurable par l'admin.

5. Gestion des comptes utilisateurs

- Chaque utilisateur a un solde qu'il peut **créditer ou débiter**.
- Un administrateur peut ajuster le solde d'un utilisateur si nécessaire.

6. Consultation des crypto-monnaies

— Tout visiteur peut consulter la liste des crypto-monnaies et leur prix en temps réel.
— Les utilisateurs connectés peuvent voir l'évolution des prix avec un historique détaillé.

7. Analyse des tendances et alertes

— Affichage de graphiques montrant l'évolution du prix des crypto-monnaies par jour/semaine/mois en utilisant **Chart.js**.
— **Système d'alerte** permettant aux utilisateurs de définir un seuil de variation.
— Option pour **vendre automatiquement** une crypto si elle atteint un seuil critique.

8. Rapport des placements globaux et exportation en PDF

— Affichage d'un **rapport détaillé des placements globaux** de l'utilisateur.
— Affichage des **achats globaux** et du **rendement global**.
— Utilisation des **couleurs rouges et vertes** pour représenter les gains et pertes.
— Possibilité d'**exporter ce rapport en PDF** pour archivage et consultation hors ligne.

9. Système de logs et audit

Implémentation d'un **système de logs** permettant de savoir :
— Quelle action a été réalisée.
— À quelle heure et date.
— À partir de quel navigateur/client Web.
— Qui a effectué l'action.

5. Sécurité et protection contre les injections SQL

— Utiliser des **requêtes préparées** avec PDO pour toutes les interactions avec la base de données.
— Valider et filtrer toutes les entrées utilisateur avant leur stockage ou utilisation.
— Implémenter des **mesures de protection contre les attaques XSS et CSRF**.

6. Rôles et responsabilités

— **Développeur Backend** : PHP (POO, sessions, requêtes SQL sécurisées, MVC).
— **Développeur Frontend** : Interface et AJAX.
— **Administrateur BDD** : Optimisation des requêtes.
— **Analyste** : Algorithmes de simulation et indicateurs techniques.

7. Rapport et évaluation

Le rapport doit inclure :
— **L'architecture de l'application** et du modèle MVC utilisé.
— **Les choix majeurs** justifiés.
— **Les fonctionnalités implémentées** et celles qui pourraient être améliorées.
— **Les points dont vous êtes fiers**.
— **Résumé des compétences acquises** par chaque membre de l'équipe.
— **Auto-évaluation** de chaque membre sur une échelle de "Peu satisfait" à "Très satisfait".

8. Barème de notation

Critère	Points
Fonctionnalités implémentées	40
Qualité du code (POO, MVC, SQL sécurisé)	20
Interface utilisateur et UX	15
Système de logs et audit	10
Rapport et justification des choix	10
Travail d'équipe et auto-évaluation	5
Total	**100**

9. Conclusion

Ce projet vous permettra de mettre en pratique toutes les compétences acquises en PHP, bases de données, AJAX et POO. Travaillez efficacement en équipe, divisez les tâches et documentez bien votre code ! Bonne chance !