Cahier de charge projet Bitchest

Sommaire

1. Présentation
2. Adresse GitHub
3. Analyse du client
4. Choix technologiques
5. L’architecture technique
6. Stratégie de sécurité
7. Déploiement
8. Méthode de travail
9. Outils utilisés
10. Evaluation du temps de travail
11. Liste fonctionnelle
12. Recettage
13. Diagramme ERP
14. Plan des écrans
15. Wireframes
16. Présentation

Nom du projet : Bitchest

Date : 23/02/24

Nom des développeurs :

* Coulibaly Oumou
* Gnakalé Jonathan
* Tiono Sandra

1. Adresse GitHub

<https://github.com/Princessai/projet_bitchest.git>

1. Analyse Client

Nous avons été recruter comme développeur par Jérôme le directeur de la startup Bitchest pour participer au développement de cette plateforme. Le but du projet est de créer une plateforme permettant aux clients de celle-ci de pouvoir acheter et vendre des crypto monnaies tel que le bitcoin et l’Ethereum. Nous étions responsables dans se projet de la partie du serveur ainsi que le prototype de l’interface graphique.

1. Choix technologiques

Les technologies beaucoup utiliser dans se projet aussi bien en front qu’en back sont Php, Laravel, Boostrap, Html et Css. Car ses languages de programmations utilisées sont parfaitement en accord avec la fluidité et la réalisation de la plateforme et aussi XD et illustrator pour la réalisation du prototype.

1. L’architecture technique
2. Stratégie de sécurité

Pour sécuriser notre application de trading, plusieurs mesures peuvent être mises en place :

* Mesures d'authentification :

Mise en place de politiques de mots de passe robustes, avec des exigences minimales de longueur et de complexité.

Architecture des parties publiques/privées :

Séparation claire entre les parties publiques accessibles aux utilisateurs et les parties privées réservées aux administrateurs et aux opérations sensibles.

* Mesures sur les mots de passe :

Stockage sécurisé des mots de passe des utilisateurs.

Utilisation de mots de passe uniques pour chaque utilisateur.

* Mesures sur la base de données :

Chiffrement des données sensibles stockées en base de données pour protéger les informations financières et personnelles des utilisateurs.

Mise en œuvre de contrôles d'accès stricts pour limiter l'accès aux données uniquement aux utilisateurs autorisés.

* Mesures sur le front-end :

Validation côté client et côté serveur des données saisies.

Utilisation de bibliothèques et de Framework de développement sécurisé pour réduire les vulnérabilités potentielles du front-end.

1. Déploiement

Déploiement de notre application de trading

Ce guide vous accompagnera dans le déploiement de notre application de trading.

Technologies à vérifier

Assurez-vous que les technologies et versions suivantes sont installées sur votre système :

PHP 8.0 ou supérieur : vous pouvez vérifier votre version PHP en exécutant php -v dans le terminal.

Composer : gestionnaire de dépendances PHP. Vérifiez la version avec composer -v.

Serveur web (par exemple Apache) avec le module PHP activé.

MySQL 8.0 ou supérieur : vérifiez la version avec mysql -v.

Laravel 8.x : la version spécifique peut être vérifiée dans le fichier composer.json.

Actions à effectuer dans le terminal

Clonez le dépôt du projet dans votre environnement de développement :

git clone <référentiel\_git\_du\_projet>

Accédez au répertoire du projet :

cd <nom\_du\_répertoire\_du\_projet>

Installez les dépendances du projet avec Composer :

composer install

Copiez le fichier .env.example et renommez-le en .env :

cp .env.example .env

Générez une clé d'application Laravel :

php artisan key:generate

Migrez la base de données :

php artisan migrate

Servez l'application avec le serveur web de votre choix.

php artisan serve

1. Méthode de travail

Méthodologie de travail en équipe

Notre équipe suit une méthodologie Agile, en particulier la méthode Scrum. Cette approche nous permet de travailler en itérations courtes, d'adapter rapidement nos plans en fonction des besoins changeants et de fournir des résultats tangibles à intervalles réguliers.

Méthodologie Agile :

Planification : Nous commençons par une réunion de planification où nous définissons les objectifs de la sprint et sélectionnons les tâches à réaliser.

Sprint : Nous travaillons sur les tâches sélectionnées pendant une période de deux semaines.

Daily Standups : Chaque jour, nous avons une réunion quotidienne de synchronisation où chaque membre de l'équipe partage ses progrès, ses défis et ses objectifs pour la journée.

Revue de Sprint : À la fin de chaque sprint, nous organisons une réunion pour passer en revue les fonctionnalités développées et recueillir les commentaires.

Rétrospective : Nous terminons chaque sprint par une réunion de rétrospective pour discuter des points forts et des points à améliorer dans notre processus de travail.

Processus de Commit :

Avant de commencer à travailler sur une nouvelle fonctionnalité ou un correctif, nous nous assurons d'être à jour avec la branche principale du dépôt.

Pendant le développement, nous faisons des commits réguliers pour enregistrer nos modifications de manière incrémentielle.

Avant de pousser nos modifications, nous vérifions que notre code respecte les normes de qualité et qu'il n'y a pas de conflits avec la branche principale.

Nous rédigeons des messages de commit clairs et descriptifs pour expliquer les modifications apportées.

Convention de Nommage des Commits :

Nous suivons une convention de nommage des commits pour faciliter la compréhension et la gestion des modifications dans notre dépôt.

1. Outils utilisés

Environnement de développement

Éditeur de code : Visual Studio Code, Sublime Text, PHPStorm

Navigateur web : Google Chrome, Mozilla Firefox, Microsoft Edge

Terminal : Git Bash (Windows)

Versionnement de code source : Git

Gestionnaire de dépendances : Composer (PHP)

Base de données locale : MySQL

Outils collaboratifs

Système de suivi des problèmes : GitHub , Jira,

Plateforme de développement collaboratif : GitHub

Visioconférence : Zoom

Outils et services

Framework PHP : Laravel

CSS Framework : Bootstrap

Services de gestion de projets : Jira

1. Evaluation du temps de travail

|  |  |
| --- | --- |
| Grand poste de développement | Nombre de jours de travail (1 journée = 7 heures) |
| Analyse des exigences et conception | 3 |
| Configuration de l'environnement de développement | 2 |
| Mise en place de l'infrastructure backend | 2 |
| Mise en place de l'infrastructure frontend | 3 |
| Implémentation des fonctionnalités de base | 1 |
| Implémentation des fonctionnalités avancées | 2 |
| Tests unitaires et d'intégration | 1 |
| Déploiement et configuration de la production | 1 |
| Documentation et formation | 2 |
| Tests et corrections de bogues | 1 |

1. Liste fonctionnelle

Fonctionnalités front-end

Inscription et connexion des utilisateurs : formulaire d'inscription, formulaire de connexion, , réinitialisation des informations

Tableau de bord utilisateur : affichage des informations de compte, des soldes, des positions ouvertes et des transactions historiques

Recherche de titres : barre de recherche pour rechercher des actions

Historique des transactions : affichage des transactions exécutées, y compris les dates, les prix, les quantités et les statuts

Portefeuille : affichage des positions actuellement détenues

Paramètres du compte : formulaire pour mettre à jour les informations personnelles, les préférences de notification et les méthodes de paiement

Fonctionnalités back-end

Authentification et autorisation : implémentation de l'inscription, de la connexion, de la vérification de l'e-mail et de la réinitialisation du mot de passe pour les utilisateurs

Gestion des utilisateurs : CRUD (création, lecture, mise à jour, suppression) pour les utilisateurs, y compris les informations personnelles, les préférences de notification et les méthodes de paiement

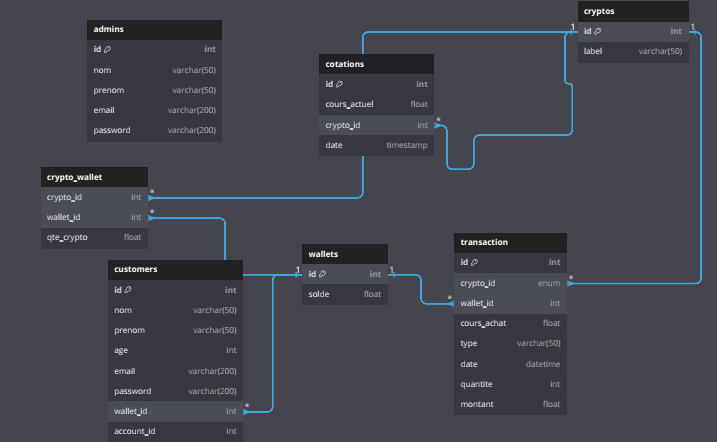
Historique des transactions : enregistrement et gestion des transactions exécutées, y compris les dates, les prix, les quantités et les statuts

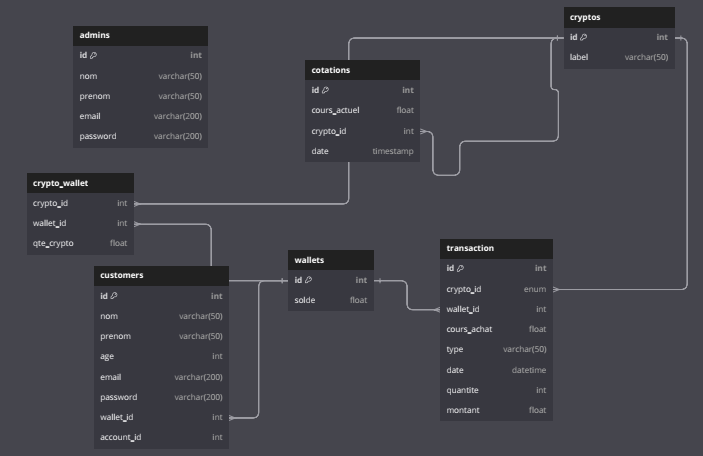
Portefeuille : gestion des positions actuellement détenues

1. Recettage

|  |  |
| --- | --- |
| Fonctionnalité | Statut |
| Inscription et connexion des utilisateurs | Opérationnelle |
| Tableau de bord utilisateur | Opérationnelle |
| Historique des transactions | Opérationnelle |
| Portefeuille | Opérationnelle |
| Paramètres du compte | Opérationnelle |
| Authentification et autorisation | Opérationnelle |
| Gestion des utilisateurs | Opérationnelle |
| Gestion des paiements | Opérationnelle |
| Tests et déploiement | Opérationnelle |

1. Diagramme REP





1. Plan des écrans

(Voir dossier ajouter au cahier de charge)

1. Wireframe

