Projet #5

BitChest

CDA 3eme année - **L'École Multimédia - 2025**

l'école multimédia

Sommaire

Contexte

Présentation

Vous venez d'être recruté comme développeur par Jérôme - directeur d'une toute nouvelle startup dénommée **BitChest** - afin de participer au développement de leur plateforme.

BitChest a pour vocation de permettre à des particuliers d'acheter et de vendre des crypto-monnaies (Cryptocurrencies), tel que le Bitcoin ou l'Ethereum.

L'objectif actuel de BitChest est de développer un prototype fonctionnel afin de lever des fonds pour développer la plateforme commerciale.

Vous êtes embauché en tant que **développeur fullstack** et serez responsable de la partie serveur ainsi que le prototype de l'interface graphique.

BitChest

BitChest a été créé par Jérôme, ancien trader et passionné par les cryptos depuis la première heure.

L'équipe actuelle est réduite : Elle est composé de Jérôme (Directeur), Lisa (Commercial) et vous (Développeur.se)

Les *exchanges* de crypto-monnaies sont nombreux, mais Jérôme veut développer une application simple et opérationnelle pour la vendre en marque blanche à des entreprises financières.

Le modèle économique sera basé sur une redevance payée par ses clients, et le partage des frais de transaction.

Charte graphique

BitChest est une toute nouvelle startup et la charte graphique est minimale :

Typographie : Celias
Couleurs : https://coolors.co/01ff19-ff5964-ffffff-38618c-35a7ff

Contraintes

Vous travaillerez **en équipe de 3. (minimum 2 personnes)**

L'application web que vous allez développer doit impérativement répondre à ces critères :

1. L'application doit utiliser le framework **Symfony** et la BDD de type **mySQL**
2. Elle doit **répondre aux demandes** décrites dans le cahier des charges
3. L'application est en **anglais**

4. Vous appliquerez la méthodologie **SCRUM** pour le développement

Libertés

1. Vous êtes libre d'utiliser ou non **des librairies CSS**
2. Vous êtes libre d'utiliser **un framework Javascript**

Déroulé du module

Vous devez réaliser le projet en parallèle des cours.
Voici un déroulé des éléments à réaliser suivant vos modules de cours

Conception logicielle avec UML

- Mise en place du projet
- Mise en place des outils (git/github, collaboratifs)
- Conception UML
- Liste fonctionnelle
- Zoning et wireframe
- Maquette Figma

Développer la partie back-end avec le framework Symfony

- Intégration du front-end
- Mise en place de la BDD
- Développement de la partie back-end
- Ecriture du journal de développement

Les tests logiciels assistés par l'IA

- Ecriture du plan de test
- Ecriture des tests unitaires
- Mise en ligne
- Finalisation des documents

Méthodologie

Vous réaliserez ce projet en employant la **méthodologie SCRUM**.

Votre formateur assurera le rôle de **Product Owner**.
Le rôle de **Scrum Master** sera tenu par l'un d'entre vous à tour de rôle.

Vous devez donc mettre en place les artefacts nécessaires (Backlog, Sprint board) et les rituels (Sprints planning, Sprints, Sprint review, Sprint rétrospective)

Pour les Sprint review et les Sprints rétrospective, le Scrum Master produira un compte-rendu de ce qui c'est dit et des réponses apportées (à ajouter au **Journal de développement**, voir Annexe)

Objectifs

A l'issue du module, vous devrez avoir réalisé les éléments suivant :

1. Le prototype de la partie front-end
2. La partie back-end de l'application, **fonctionnelle et en ligne**

Rendu final

Votre rendu final prendra la forme **d'une archive Zip** et devra comporter les éléments suivants :

1. Le dossier de conception
2. Le journal de développement
3. Le code de votre application

Vous devez livrer une archive de votre livrable avec l'ensemble des éléments, à l'exception des dossiers suivant :

- Node_modules
- vendors

Cette archive aura comme titre le nom du groupe suivi de votre classe.
Exemple: **`groupeA_projet4_CDA3.zip`**

Attention : Un rendu non livré ou en retard vous pénalise pour la certification (non validation des compétences RNCP)

Conseils

- Bien prendre le temps d’analyser le brief et comprendre le client
- Organisez-vous et planifiez votre travail : donnez vous des objectifs intermédiaires
- Planifiez des sessions de travail régulière pour votre équipe
- Ne jamais être trop ambitieux
- Faites directement des documents de conception présentables
- Complétez les documents à rendre au fur et à mesure
- Mettez en oeuvre les bonnes pratiques vues en cours
- Refactoriser pour éviter le code redondant
- Soignez la qualité de votre code (commentaires, indentation, nommage)
- N’attendez pas la fin pour commencer à travailler sur votre projet
- Pensez à la facilité d’utilisation et la qualité du résultat !

Compétences à valider

La notation de votre projet se fera en évaluant les compétences de la certification regroupées ci dessous :

Compétence 3 : Développer des composants métier

Les bonnes pratiques de la programmation orientée objet (POO) sont respectées
Les composants métier sont sécurisés
Les règles de nommage sont conformes aux normes de qualité de l'entreprise
Le code source est documenté
Les traitements répondent aux fonctionnalités décrites dans le dossier de conception
Les tests unitaires sont réalisés
Les tests de sécurité sont réalisés
La démarche structurée de résolution de problème est adaptée en cas de dysfonctionnement
Le système de veille permet de suivre les évolutions technologiques et les problématiques de sécurité en lien avec les composants métier d'une application

Compétence 4 : Contribuer à la gestion d'un projet informatique

Les tâches de conception et de développement sont planifiées en fonction du délai défini
Le suivi des tâches est mis en rapprochement avec la planification, les éventuels retards sont identifiés et les acteurs concernés sont alertés
Les procédures qualité sont mises en oeuvre
L'environnement de développement défini est en adéquation avec l'architecture du projet
Les outils collaboratifs sont choisis en fonction de la méthode de développement
Les comptes rendus de réunion sont structurés, rédigés dans un style adapté, dans le respect des règles orthographiques et grammaticales, et contiennent les informations nécessaires

Compétence 6 : Définir l'architecture logicielle d'une application

L'architecture logicielle est conforme aux bonnes pratiques d'une architecture multicouche répartie sécurisée
Le rôle de chaque couche est bien défini en tenant compte de la stratégie de sécurité
Les besoins d'éco-conception de l'application sont identifiés

Compétence 7 : Concevoir et mettre en place une base de données relationnelle

Le schéma conceptuel respecte les règles du relationnel
Le schéma physique est conforme aux besoins exprimés dans le cahier des charges
Les règles de nommage ont été respectées
L'intégrité, la sécurité et la confidentialité des données est assurée
La base de données de test est créée avec un jeu d'essai complet et peut être restaurée en cas d'incident
La documentation technique des bases de données est comprise, en langue française ou anglaise (niveau B1 du CECRL pour l'anglais)

Compétence 8 : Développer des composants d'accès aux données SQL et NoSQL

Les traitements relatifs aux manipulations des données répondent aux fonctionnalités décrites dans le dossier de conception
Les cas d'exception sont pris en compte
L'intégrité et la confidentialité des données sont maintenues
Les conflits d'accès aux données sont gérés
Toutes les entrées sont contrôlées et validées dans les composants serveurs sécurisés
Les tests unitaires et de sécurité sont associés à chaque composant
La démarche structurée de résolution de problème est adaptée en cas de dysfonctionnement
Le système de veille permet de suivre les évolutions technologiques et les problématiques de sécurité liées aux bases de données SQL et NoSQL

Compétence 9 : Préparer et exécuter les plans de tests d'une application

Le plan de tests couvre l'ensemble des fonctionnalités retenues pour l'application
Un environnement de tests est créé
L'intégralité des tests exécutés sont conformes au plan de tests défini
Les résultats obtenus sont cohérents avec les résultats attendus
Le plan de tests tient compte des évolutions technologiques et des problèmes de sécurité liés aux tests logiciels

Evaluation lors du suivi de projet

Les compétences suivantes seront évaluées pendant le temps de suivie de projet :

- Compétence 4

Les éléments indispensables

- Documenter le code en anglais
- Mettre en place une stratégie de sécurité forte
- Versionner votre progression avec Git

Gestion du repo git en équipe

Comme vous allez travailler en équipe, il est important que chaque élève fasse ses propres commits réguliers.
Chaque élève doit intervenir sur l'ensemble de l'application (conception, front, back, BDD)

Le correcteur regardera les commits pour constater la participation active de chaque élèves

Le dossier de conception

Ce document que vous devez réaliser regroupe l'ensemble des éléments de conception que vous devez réaliser.

Structure

1. Reformulation de la demande client
2. Adresse Github
3. Les documents de conceptions de l'interface
    a. Zoning
    b. Wireframe
    c. Maquette Figma
    d. Schémas de l'enchaînement des écrans
4. La liste fonctionnelle
5. Les schémas de la base de données
6. L'architecture technique
7. Les besoins d'éco-conception de l'application
8. La stratégie de sécurité
9. Le plan de tests
10. Les recherches sur internet
    a. Résolution de problèmes
    b. Veille technique
11. Des captures de chaque écrans (screenshots)
12. Le déploiement

Note : La maquette Figma doit être publique et accessible par le correcteur

L'architecture technique

Faites un schéma de l'architecture de l'application, avec comme référence :

- La couche présentation
- La couche métier (les composants implémentant les fonctionnalités métiers)
- La couche sécurité
- La couche gérant les données

La stratégie de sécurité

Décrivez les règles mises en place pour sécuriser :

- Le front
- Le back
- La base de données

Le plan de test

Votre plan de test complet ou un OPTP (One Page Test Plan)

Le déploiement

Écrire une note permettant au correcteur de déployer votre projet à partir du code source livré :

- Technologies à vérifier
- Actions à effectuer dans le terminal
- Toutes autres actions nécessaires pour arriver au rendu final

Le journal de développement

Ce document comporte les comptes rendus réalisés pendant le développement.
Il prendra la forme d'un document textuel. (.doc)

Structure

Il doit au minimum comporter les sections suivantes :

Les comptes rendu de développement

Lors du développement, des difficultés vont émerger : difficulté à réaliser une fonctionnalité, bugs, problèmes d'installation, etc …

C'est ici que vous consignez ces difficultés, et les réponses apportées.

Les comptes rendus des Sprints

Les rituels SCRUM (Sprint review et les Sprints rétrospective) produiront des questions, des réponses et des conclusions.
C'est ici que le Scrum Master consignera ces compte-rendus.

Les comptes rendu des tests utilisateurs

Lors de la création de la maquette graphique de l'interface, vous devez faire tester celle-ci par des utilisateurs externes (choisissez des élèves de votre classe).

Vous noterez leurs retours et vos conclusions.

La demande client

Le produit est une **application web** utilisable avec un navigateur web et en **langue anglaise**

Elle doit **s'adapter au périphérique** comme toute application moderne (mobile, tablette, desktop, paysage et portrait)

Les administrateurs

Ce sont les agents de **BitChest** qui vont gérer le site au jour le jour avec Jérôme.
Une fois identifiés, ils doivent pouvoir réaliser les actions suivantes :

Modifier leur donnée personnelle

Gérer les clients
Création, Affichage, Modification et Suppression d'un utilisateur.
Les mots de passe ne peuvent être modifiés ou visualisés.

Lors de la création d'un nouvel utilisateur, un mot de passe temporaire est généré et présenté à l'administrateur.
Celui-ci peut l'envoyer à l'utilisateur, qui pourra le modifier lorsqu'il accédera à son interface d'administration privée.

Lors de la création d'un nouvel utilisateur, **le compte est crédité de 500 euros** pendant la phase de prototypage.

Consultation des cours des crypto monnaies
Affichage de la liste des cryptos et de leur cours actuel.

Les clients

Ce sont les clients de **BitChest**. Via leur interface d'administration privée, ils peuvent effectuer les actions suivantes:

Gérer leur donnée personnelle

Gérer le portefeuille
Affichage du contenu du portefeuille
Affichage du solde en euro (doit être toujours visible)
Vente d'une crypto monnaie

Consultation des cours des crypto monnaies

Affichage de la liste des cryptos et de leur cours actuel
Consultation de la courbe de progression de chaque crypto monnaies
Achat d'une quantité de crypto monnaies au cours actuel

Le portefeuille

Chaque client possède un portefeuille privé regroupant l'ensemble de ses achats en crypto-monnaies.

Le client peut donc voir:

- La liste des crypto monnaies qu'il possède
- Pour chaque crypto, les différents achats qu'il a effectués (date, quantité et cours)
- Pour chaque crypto, la valeur d'achat d'une unité
- Pour chaque crypto, la plus-value actuelle (gain en cas de vente totale)

Exemple :

Bruno a fait plusieurs achats de BTC :

- 1 BTC à 10 000 euros le 01/10/2020
- 0.5 BTC à 18 000 euros le 14/03/2021
- 0.5 BTC à 20 000 euros le 27/01/2022

Pour le BTC, la valeur d'achat d'une unité est calculée comme suite :

- Cout total : (1 x 10 000) + (0.5 x 18 000) + (0.5 x 20 000) = **29 000 euros**
- Quantité possédée : 1 + 0.5 + 0.5 = **2 BTC**
- Valeur d'achat : 1 BTC = 29 000 / 2 soit **14 500 euros**

Si le cours actuel du BTC est à **30 000 euros**, la plus-value actuelle de son portefeuille BTC est calculée comme suite :

- Valeur totale au cours actuel : (1 + 0.5 + 0.5) x 30 000 = **60 000 euros**
- Plus value actuelle: 60 000 - 14 500 euros = **45 500 euros**

La plus value actuelle peut être négative (en perte)

La vente

Si le client décide de vendre, il récupère - en euro - la valeur de son achat.
Les calculs de valeur d'achat et de plus value doivent prendre en compte le nouveau solde crypto.

Les crypto monnaies

Les 10 crypto monnaies qui seront gérées par Bitchest lors du lancement sont les suivantes :

1. Bitcoin
2. Ethereum
3. Ripple
4. Bitcoin cash
5. Cardano
6. Litecoin
7. Dash
8. Iota
9. NEM
10. Stellar

Afin de pouvoir tester la plateforme lors de la phase de prototypage, vous devez générer les cotations des dix crypto-monnaies supportées sur les 30 derniers jours.

Pour ce faire, vous utiliserez la fonction php se trouvant dans le document *cotation_generator.php*

La première cotation sera quand à elle générer par une autre fonction qui se trouve également dans le document *cotation_generator.php*

Attention, un cours de crypto-monnaie ne peut être négatif.

Afin de pouvoir voir l'évolution du cours d'une crypto monnaie, vous devez afficher son évolution sous forme graphique.

A vous de choisir la solution javascript la plus adaptée.

Les ressources

https://drive.google.com/file/d/1sANgMFR58lWFD30EfuPQcnVCph_vBfkN/view?usp=sharing

La page d'inscription

Cette est public et permet à un utilisateur anonyme de s'inscrire à BitChest en transmettant un mail et un mot de passe (double vérification)

L'interface d'administration

Login

Afin d’accéder à l’interface, les administrateurs comme les clients se connectent sur la même page de login public.

Grâce à une adresse mail et à un mot de passe, ils peuvent accéder à leur espace privé respectif.

Interface privée

Vous êtes libre de structurer le contenu comme bon vous semble.
L’unique contrainte est que toutes les fonctionnalités demandées soient accessibles.

Veillez cependant à ce que ce soit lisible : **le but ici est de faire un prototype d’interface clair et fonctionnel.**