Projet #1



Bachelor année 3 - **L'École Multimédia**





Sommaire

[Contexte](#_33qwhkwn1rch)

[Présentation](#_sailtqthv67f)

[BitChest](#_j42nkfbj0h1x)

[Charte graphique](#_diqhvnnnb170)

[Contraintes](#_vdoag9javd0m)

[Libertés](#_xaoaufvcb0e6)

[Méthodologie](#_es3nn0843qbj)

[Objectifs](#_jrzlmxp46c1v)

[Cahier des charges](#_x6ekr5eft81z)

[Les administrateurs](#_9tk1s2cucdcd)

[Modifier leur donnée personnelle](#_3mc9ix8yh96r)

[Gérer les clients](#_3mc9ix8yh96r)

[Consultation des cours des crypto monnaies](#_371ifxfjzlis)

[Les clients](#_ei2lj6tq61g)

[Gérer leur donnée personnelle](#_2ymela3dway)

[Gérer le portefeuille](#_2ymela3dway)

[Consultation des cours des crypto monnaies](#_v6qe8vhc2vb9)

[Le portefeuille](#_xp7qs3933xvt)

[Exemple :](#_sdinzots1sno)

[La vente](#_co2i68naqs00)

[Les crypto monnaies](#_gg86svwazgg)

[L’interface d'administration](#_i0yxamgzw4y5)

[Login](#_1ldrbacaikhn)

[Interface privée](#_k4ohj6vt92mr)

[Rendu final](#_f7rp9tx258ep)

[Conseils](#_2cjdsgbbn6ex)

[Compétences à valider](#_ebanhl79ia2s)

[C1. Maquetter une application](#_1656i13uuk9c)

[C4. Développer la partie front-end d’une interface utilisateur web](#_lwci1cmyjnhy)

[C3. Développer des composants d’accès aux données](#_l08v9av9ua62)

[C5. Développer la partie back-end d’une interface utilisateur web](#_ad5x02jf2kta)

[C6. Concevoir une base de données](#_nadmiw5pro5e)

[C7. Mettre en place une base de données](#_bzaimmb5caqu)

[C8. Développer des composants dans le langage d’une base de données](#_76z2s0eg7x2t)

[C9. Collaborer à la gestion d’un projet informatique et à l’organisation de l’environnement de développement](#_c9aew614s0eg)

[C10. Concevoir une application](#_2laeufr772ld)

# 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**
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**

# 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

# Cahier des charges

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. NEM
8. Stellar
9. IOTA
10. Dash

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.

## **L’interface d'administration**

### **Login**

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

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



### **Interface privée**

L’administration est structurée suivant le zoning suivant.

La colonne A renferme les diverses actions que peut effectuer l’utilisateur.

La zone B renferme le contenu de chaque page.



Attention de bien appliquer les même typologie d’interface aux même fonctionnalités, afin d’apporter de la cohérence à votre interface

Vous êtes libre de structurer le contenu comme bon vous semble. Veillez cependant à ce que ce soit lisible : **le but ici est de faire un prototype d’interface claire et fonctionnel.**

# Rendu final

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

1. Le cahier des charges technique (voir Annexe)
2. Le journal de développement (voir Annexe)
3. Le code de votre application
4. Un lien vers votre Sprintboard et Backlog

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: Bitchest\_groupe\_A\_CDA.zip

# 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
* 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 :

Note : Les compétences en bleu pourront être validées ultérieurement

## C1. Maquetter une application

|  |
| --- |
| La maquette prend en compte les spécificités fonctionnelles décrites dans les cas d'utilisation ou les scénarios utilisateur |
| L'enchaînement des écrans est formalisé par un schéma |
| La maquette et l'enchaînement des écrans sont validés par l’utilisateur final |
| La maquette respecte la charte graphique de l’entreprise |
| La maquette est conforme à l'expérience utilisateur et à l’équipement ciblé |
| La maquette respecte les principes de sécurisation d’une interface utilisateur |
| La maquette prend en compte les exigences de sécurité spécifiques de l’application |
| La communication écrite en français ou en anglais est rédigée de façon adaptée à l’interlocuteur et sans faute |

## C4. Développer la partie front-end d’une interface utilisateur web

|  |
| --- |
| Les pages web s’adaptent à la taille de l’écran et sont fluides |
| Le code source est documenté ou auto-documenté |
| Les tests garantissent que les pages web répondent aux fonctionnalités décrites dans le dossier de  conception technique |
| Les tests de sécurité suivent un plan reconnu par la profession |
| L’objet de la recherche est exprimé de manière précise en langue française ou anglaise |
| La documentation technique liée aux technologies associées, en français ou en anglais, est comprise  (sans contre-sens, ...) |
| La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une  nouvelle fonctionnalité |
| La veille sur les vulnérabilités connues permet d’identifier et corriger des failles potentielles |
| Le partage du résultat de veille est effectué oralement ou par écrit avec ses pairs |

## C3. Développer des composants d’accès aux données

|  |
| --- |
| Les traitements relatifs aux manipulations des données répondent aux fonctionnalités décrites dans le  dossier de conception technique |
| Un test unitaire est associé à chaque composant, avec une double approche fonctionnelle et sécurité |
| Le code source des composants est documenté ou auto-documenté |
| Les composants d’accès à la base de données suivent les règles de sécurisation reconnues |
| La sécurité des composants d’accès se fonde sur les mécanismes de sécurité du SGBD |
| L’objet de la recherche est exprimé de manière précise en langue française ou anglaise |
| La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité |
| La veille sur les vulnérabilités connues permet d’identifier et corriger des failles potentielles |
| La documentation technique liée aux technologies associées, en français ou en anglais, est comprise (sans contre-sens,...) |
| Le partage du résultat de veille est effectué oralement ou par écrit avec ses pairs |

## C5. Développer la partie back-end d’une interface utilisateur web

|  |
| --- |
| Les bonnes pratiques de développement objet sont respectées |
| Les composants serveur contribuent à la sécurité de l’application |
| Le code source des composants est documenté ou auto-documenté |
| Les tests garantissent que les traitements serveurs répondent aux fonctionnalités décrites dans le dossier de conception technique |
| Les tests de sécurité suivent un plan reconnu par la profession |
| L’objet de la recherche est exprimé de manière précise en langue française ou anglaise |
| La documentation technique liée aux technologies associées, en français ou en anglais, est comprise (sans contre-sens, ...) |
| La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité |
| La veille sur les vulnérabilités connues permet d’identifier et corriger des failles potentielles |
| Le partage du résultat de veille est effectué oralement ou par écrit avec ses pairs |

## C6. Concevoir une base de données

|  |
| --- |
| Le schéma entité-association des données couvre les règles de gestion sur les données |
| Le schéma entité-association des données respecte le formalisme du modèle entité-association |
| Les règles de nommage sont conformes aux normes qualité de l'entreprise |
| Le schéma physique de la base de données est normalisé |

## C7. Mettre en place une base de données

|  |
| --- |
| La base de données relationnelles est conforme au schéma physique |
| Les règles de nommage sont conformes aux normes qualité de l'entreprise |
| L'intégrité des données est assurée |
| La base de données est disponible avec les droits d'accès prévus |
| Le niveau de confidentialité demandé est respecté |
| Les utilisateurs sont authentifiés et leurs actions peuvent être tracées |
| La base de données de test peut être restaurée en cas d'incident |
| L’objet de la recherche est exprimé de manière précise en langue française ou anglaise |
| La démarche de recherche permet de trouver une solution à un problème technique ou à la mise en œuvre d’une nouvelle fonctionnalité |
| La veille sur les vulnérabilités connues permet d’identifier et corriger des failles potentielles |
| La documentation technique liée aux technologies associées, en français ou en anglais, est comprise (sans contre-sens,...) |
| La communication écrite, en français ou en anglais, est rédigée de façon adaptée à l’interlocuteur et sans faute |
| Le partage du résultat de veille est effectué oralement ou par écrit avec ses pairs |

## C8. Développer des composants dans le langage d’une base de données

|  |
| --- |
| Les traitements relatifs aux manipulations des données répondent aux fonctionnalités décrites dans le dossier de spécifications |
| 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 |
| Un test unitaire est associé à chaque composant, avec une double approche fonctionnelle et sécurité |

## C9. Collaborer à la gestion d’un projet informatique et à l’organisation de l’environnement de développement

|  |
| --- |
| Le suivi des activités ou des tâches du projet est mis en œuvre en fonction de la démarche projet adoptée |
| Les procédures qualité décrites dans le plan qualité projet sont mises en œuvre |
| L'environnement de développement est défini |
| Les outils collaboratifs sont choisis |
| La communication écrite en français ou en anglais est rédigée de façon adaptée à l’interlocuteur et sans faute |
| La communication technique orale, en français ou en anglais, est réalisée de façon adaptée à  l’interlocuteur et sans faute |

## C10. Concevoir une application

|  |
| --- |
| Les cas d’utilisation couvrent l'ensemble des exigences utilisateur exprimées dans le cahier des charges |
| Les besoins de sécurité de l’application sont identifiés |
| Les besoins d'éco conception de l’application sont identifiés |
| Les classes d’analyse et de conception sont définies |
| L’architecture technique est conforme aux bonnes pratiques d’une architecture répartie sécurisée |
| Le dossier de conception est structuré et documenté en conformité avec la démarche choisie |
| Le rôle de chaque couche dans la stratégie de sécurité est bien défini |
| L’objet de la recherche est exprimé de manière précise en langue française ou anglaise |
| La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité |
| La documentation technique liée aux technologies associées, en français ou en anglais, est comprise (sans contre-sens, ...) |

# ANNEXES

# Cahier des charges technique

A partir de ce brief, vous devez rédiger un cahier des charges technique qui encadre votre développement.

Il prendra la forme d’un document textuel (.doc)

Il doit au minimum comporter les sections suivantes :

## Présentation

Nom du projet, date, noms des développeurs

## Adresse Github

L’adresse Github du projet

## Analyse client

Reformuler ici avec vos mots votre compréhension du projet - votre client - et des implications techniques que cela engendre.

## Choix technologiques

Listez les technologies employées, et motivez vos choix technologiques, aussi bien côté front que back

## L’architecture technique

Faites un schéma de l’architecture de l’application, avec les clients front et la structure back (serveur, base de données, échanges front/back …)

## Stratégie de sécurité

Décrivez ici les mesures mises en place pour sécuriser l’application :

* Mesures d’authentification
* Architecture des parties publics/privées
* Mesures sur les mots de passe
* Mesures sur la base de données
* Mesures sur le front-end

## Déploiement

Écrire une note permettant à un développeur 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

## Méthode de travail

Décrivez ici votre méthodologie de travail en équipe : waterfall, agile.

Votre process pour chaque commit, votre convention de nommage des commits

## Outils utilisés

Vous écrirez ici une liste exhaustive des outils utilisés durant le développement :

* L’environnement de développement
* Les outils collaboratifs
* Outils et services utilisés

## Evaluation du temps de travail

Faites un petit tableau avec comme entrée chaque grand poste de développement, et comme sortie le nombre de jour de travail (1 journée = 7 heures)

## Liste fonctionnelle

Listez de manière exhaustive les fonctionnalités à développer (front et back)

## Recettage

Faites un tableau avec comme entrée la liste fonctionnelle, et comme sortie si la fonctionnalité est opérationnelle ou non opérationnelle (ou possède un bug, une limitation …)

## Diagramme ERP de la base de données

Insérez ici un diagramme ERP de vos entités (réalisé [dbdiagram.io](https://dbdiagram.io/home) ou [mySql Worksbench](https://www.mysql.com/products/workbench/))

Dans ce but, expliciter les relations dans votre schéma : représenter les à l'aide de "flèches" normalisées entre les tables ou entités.

## Plan des écrans

Un arbre hiérarchique de chaque écran composant l’application, avec leurs liens.

## Wireframes

Les images des wireframes que vous avez réalisé lors de la préparation de votre travail. Chaque page doit avoir son wireframe.

# Le journal de développement

Ce document comporte les comptes rendus réalisés pendant le développement.

Il prendra la forme d’un document textuel. (.doc)

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 consignerez ces difficultés, et les réponses apportées.

## 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.

## 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.