

UNIVERSITÉ NATIONALE DU VIETNAM, HANOÏ
INSTITUT FRANCOPHONE INTERNATIONAL

MAYABA Solim Wapo

**Mise en œuvre d'un Smart Contract d'une
Blockchain Ethereum pour créer une
cryptomonnaie interne**

Xây dựng một giải pháp hợp đồng thông minh công nghệ Blockchain
Ethereum cho một loại tiền mã hóa nội bộ

MÉMOIRE DE FIN D'ÉTUDES DU MASTER
INFORMATIQUE

Sous la Direction de l'Ingénieur :
M. PELLAT Dominique (Architecte Bancaire)

Je approuve la soutenance.
Dominique PELLAT



HANOÏ - 2019

UNIVERSITÉ NATIONALE DU VIETNAM, HANOÏ
INSTITUT FRANCOPHONE INTERNATIONAL

MAYABA Solim Wapo


**Mise en œuvre d'un Smart Contract d'une
Blockchain Ethereum pour créer une
cryptomonnaie interne**

Xây dựng một giải pháp hợp đồng thông minh công nghệ Blockchain
Ethereum cho một loại tiền mã hóa nội bộ

Spécialité : Systèmes Intelligents et Multimédia
Code : Programme pilote

MÉMOIRE DE FIN D'ÉTUDES DU MASTER
INFORMATIQUE

Sous la Direction de l'Ingénieur :
M. PELLAT Dominique (Architecte Bancaire)

J'approuve la soutenance.
Dominique PELLAT


HANOÏ - 2019

Attestation sur l'honneur

J'atteste sur l'honneur que ce mémoire a été réalisé par moi-même et que les données et les résultats qui y sont présentés sont exacts et n'ont jamais été publiés ailleurs. La source des informations citées dans ce mémoire a bien été précisée.

LỜI CAM ĐOAN

Tôi cam đoan đây là công trình nghiên cứu của riêng tôi. Các số liệu, kết quả nêu trong Luận văn là trung thực và chưa từng được ai công bố trong bất công trình nào khác. Các thông tin trích dẫn trong Luận văn đã được chỉ rõ nguồn gốc.



MAYABA Solim Wapo

Remerciements

Je voudrais tout d'abord adresser toute ma gratitude à M. Dominique Pellat pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont largement contribué à alimenter ma réflexion. Tous les outils nécessaires, des échanges ainsi que des formations étaient mise à ma disposition afin d'obtenir des résultats significatifs. J'ai eu également le privilège de présenter ce projet à la 6 ème journée de l'innovation du Groupe La Poste. Merci de m'avoir fait confiance et permis de faire mes premières expériences en entreprise en matière de recherche.

J'aimerais aussi gratifier les efforts de M. NONNON David, architecte sécurité à la DISFE, qui a eu l'amabilité de répondre à mes questions et de fournir les explications nécessaires sur mon projet. Il m'a beaucoup appris sur les défis qu'un développeur doit relever au quotidien.

Je suis également reconnaissante envers les professeurs de l'IFI , qui m'ont fourni les outils nécessaires à la réussite de mon cursus. Je tiens à remercier spécialement Mr Vinh , qui fut le premier à me faire découvrir cette technologie "La Blockchain" qui devint la thématique de mon sujet de stage.

Je tiens également à remercier ma famille pour sa patience et bienveillance à mon égard. Merci de m'avoir encouragé et de n'avoir ménager aucun effort pour cet aventure depuis le Vietnam jusqu'en France.

MAYABA Solim Wapo

Résumé

Ce travail de recherche conduit à la réalisation pratique d'un projet de la DISFE. Ce projet vise à mettre en oeuvre des Smart Contrats pour créer une cryptomonnaie interne appelé I3Coin abrégé en 'I3C'. Basée sur la Blockchain Ethereum, cette cryptomonnaie est destinée à valoriser : l'engagement citoyen soutenu par le Groupe La Banque Postale (GLBP) et les actions facilitant la vie quotidienne dans l'entreprise. Dans ce travail, est expliqué ce qu'est la Blockchain et son fonctionnement, mais aussi ce qu'est une crypto-monnaie et un Smart Contract. Il apportera les outils et les connaissances nécessaires dans le but de pouvoir créer, soimême, sa propre crypto-monnaie ou toute autre application décentralisée.

Mots clés : blockchain, crypto-monnaie, smart contract, Ethereum, ERC20, Dapp

Abstract

This research work leads to the practical realization of a DISFE project. This project aims to implement Smart Contrats to create an internal cryptocurrency called I3Coin abbreviated as 'I3C'. Based on the Ethereum Blockchain, this cryptocurrency is designed to enhance : citizen engagement supported by the Group Postal Bank (GLBP) and actions facilitating daily life in the company. In this work, is explained what the Blockchain and its operation, but also what is a cryptocurrency and a Smart Contract. He will provide the necessary tools and knowledge to create his own cryptocurrency or any other decentralized application.

Keywords : blockchain, cryptocurrency, smart contract, Ethereum, ERC20, Dapp

Table des matières

Table des figures	vi
1 Introduction Générale	1
2 Présentation de la structure d'accueil	3
2.1 Le Groupe La Poste	3
2.2 La DISFE	5
2.2.1 Présentation	5
2.2.2 Organisation	5
3 Contexte d'étude et problématique	7
3.1 Projet iCubeCoin (I3C)	7
3.2 Objectif	7
3.3 Identification des besoins	8
3.4 Domaine d'étude	9
3.5 Spécification des technologies	9
3.5.1 Qu'est-ce qu'une Blockchain?	9
3.5.1.1 Définition générale	9
3.5.1.2 Caractéristiques essentielles d'une Blockchain	10
3.5.1.3 Fonctionnement d'une Blockchain	10
3.5.2 Qu'est-ce qu'une crypto-monnaie?	11
3.5.3 Mécanismes de consensus	12
3.5.3.1 Proof of Work (PoW)	12
3.5.3.2 Proof of Stake (PoS)	13
3.5.4 Qu'est-ce qu'Ethereum?	14
3.5.5 Qu'est ce qu'un Smart Contract?	14
3.6 Problématique	15
3.7 Résultat attendu	15
4 État de l'art	16
4.1 L'histoire des cryptomonnaies	16

4.2	Les cryptomonnaies sur le marché	16
4.2.1	Classification des cryptomonnaies sur le marché	16
4.2.2	Bitcoin (BTC)	17
4.2.3	Ethereum (ETH)	18
4.2.4	Ripple (XRP)	18
4.2.5	Bitcoin Cash (BCH)	18
4.2.6	Litecoin (LTC)	18
4.3	4 méthodes différentes pour créer sa propre cryptomonnaie	19
4.3.1	Créer sa propre blockchain	19
4.3.2	Forker une crypto-monnaie existante	19
4.3.3	Créer un jeton	20
4.3.4	Utiliser un service de création de crypto-monnaie	21
4.4	Réseau blockchain : "Mainnet & Testnets"	21
4.4.1	Testnets	21
4.4.2	Mainnet	22
4.5	Analyse et Solutions possibles	22
5	Solution proposée	24
5.1	Présentation de notre solution	24
5.2	Réseau privé	24
5.3	Jeton ERC-20	24
5.4	Logique métier de notre projet	25
5.4.1	Les acteurs	25
5.4.2	Les cas d'utilisation	26
5.4.3	Les fonctionnalités de notre application	26
5.5	DApp	27
5.6	Web3.js	28
5.6.1	Interaction avec les SmartContracts	28
5.6.2	Créer un portefeuille Ethereum avec Web3.js	28
5.7	Outils	28
5.8	Informations de test	29
5.9	Environnement de travail	29
6	Implémentations et Résultats	30
6.1	Implémentations	30
6.1.1	Installation de l'environnement de développement	30
6.1.1.1	Geth	30
6.1.1.2	NodeJS and NPM	30
6.1.1.3	Truffle	30
6.1.1.4	Atom	31
6.1.1.5	Git	31
6.1.1.6	React Developer Tools	31
6.1.2	Créer une blockchain privée	31

6.1.2.1	Création d'un bloc de genèse	31
6.1.2.2	Initialiser un noeud privé	33
6.1.2.3	Créer un compte	33
6.1.2.4	Lancer notre blockchain	34
6.1.3	Lancement de notre propre jeton ERC20 : I3Coin (utilisation de OpenZeppelin)	34
6.1.4	Implantation de la logique métier	35
6.1.4.1	Variables et types de données	35
6.1.4.2	Les fonctions de notre contrat	37
6.1.5	Lancement de notre smart contract	39
6.1.6	Conception - Coté Client	40
6.1.6.1	Template ReactJS	40
6.1.6.2	Configurer web3.js	40
6.1.7	Lancement de l'application	41
6.1.8	Difficultés rencontrées	41
6.1.8.1	L'authentification par adresse mail et mot de passe	41
6.1.8.2	Payer les transactions avec les éthers	41
6.2	Résultats	42
6.2.1	Lancement de la blockchain	42
6.2.2	Présentation de notre application	42
6.2.2.1	Authentification	42
6.2.2.2	Création d'un compte	43
6.2.2.3	Menu	44
6.2.2.4	Accueil	45
6.2.2.5	Profil utilisateur	46
6.2.2.6	Mes Projets	47
6.2.2.7	Gestion des Projets	51
6.2.2.8	Gestion kiosque (Uniquement visible pour le service RH)	54
6.2.2.9	Kiosque	56
6.2.2.10	Transfert	57
7	Conclusion et Perspectives	59
7.1	Conclusion générale	59
7.2	Perspectives	60
A	Quelques fonctions du smart contract	62
A.1	63

Table des figures

2.1 Les 5 branches métiers du Groupe La Poste	4
4.1 Code source Bitcoin sur Github	20
4.2 Tableau comparatif des crypto-monnaies	22
5.1 Structure de notre dApp	27
5.2 Communication entre la plateforme et les Smart Contracts	28
6.1 Contenu du fichier I3Coin.js	32
6.2 fichier script pour le lancement de la blockchain	34
6.3 Création du token ICubeCoin	35
6.4 Déclaration d'une variable de type énumération	36
6.5 Déclaration des structures du projet	36
6.6 Déclaration des variables de type mapping	36
6.7 Déclaration des évènements du projet	37
6.8 Constructeur du projet	38
6.9 Fonctions du projet	39
6.10 Configuration du réseau de l'exécution du smart-contract	40
6.11 Configuration de web3.js	41
6.12 Lancement de la blockchain	42
6.13 page d'authentification	43
6.14 page de création d'un compte	44
6.15 Menu à gauche des pages	45
6.16 page d'accueil	46
6.17 page - profil utilisateur	47
6.18 page - liste des projets	48
6.19 page - création d'un nouveau projet	49
6.20 page - Faire un tip sur un projet	50
6.21 page - Solutionner un projet	51
6.22 page - gestion des projets	52
6.23 page - valider projet	53

TABLE DES FIGURES

6.24 page - liste des favoris	54
6.25 page - vente des produits/services (kiosque)	55
6.26 page - ajouter des produits/services (kiosque)	56
6.27 page - achat des produits/services (kiosque)	57
6.28 page - liste des transferts I3C effectués	57
6.29 page - transférer des I3C	58
A.1 Annexe- fonctions du smart contract (a)	63
A.2 Annexe- fonctions du smart contract (b)	64

Liste des sigles et acronymes

DISFE	<i>Direction de l'Informatique des Services Financiers et de l'En-seigne La Poste</i>
LBP	<i>La Banque Postale</i>
dApp	<i>Dee-app (application décentralisée)</i>
ICO	<i>Initial Coin Offering</i>

Introduction Générale

Depuis quelques années, nous assistons à l'émergence de monnaies d'un genre nouveau, reposant sur des procédés cryptographiques, gérées en pair à pair selon un consensus distribué. La plus représentative d'entre elles, le Bitcoin, est lancée après la crise financière de 2008 . Ces crypto-monnaies viennent heurter la conception traditionnelle de la monnaie : unitaire, souveraine, territoriale et centralisée.

Le sujet sur les cyptomonnaies porte encore plus d'intérêt aujourd'hui avec l'arrivée d'un géant dans l'espace des cryptomonnaies. En effet, le géant des réseaux sociaux Facebook a annoncé le mardi 18 juin le lancement pour 2020 de "Libra" une nouvelle monnaie dématérialisée développée en partenariat avec une vingtaine d'institutions financières, comme Visa, Mastercard ou encore PayPal. Chaque information sur le sujet fait l'objet de beaucoup d'attention : Facebook deviendrait il le nouveau maître des cryptomonnaies?

C'est une occasion de revenir à la fois sur le concept des monnaies et de s'interroger sur la place de cette technologie sur laquelle se reposent les cryptomonnaies : la Blockchain. La blockchain est ainsi vue comme un potentiel successeur à la technologie existante. Ethereum est actuellement entrain d'exploser et de révolutionner la Blockchain en introduisant de nouvelles possibilités. Avec Ethereum il est possible, non seulement de réaliser de simples paiements comme le proposait déjà de faire ce que la monnaie Bitcoin proposait déjà, mais offre également la possibilité de profiter des nœuds du réseau pour exécuter des programmes, appelés des Smart Contracts. Ceux-ci permettent de créer des applications basées sur un système décentralisé. Un grand nombre de développeurs se sont alors mis à étudier Ethereum et les Smart Contracts, en participant activement à la prospérité de cette communauté.Des secteurs tels que les dossiers médicaux, la logistique, la finance, les assurances peuvent tous bénéficier des contrats intelligents visant à éliminer les tiers.

Le domaine d'application des Smart Contracts est beaucoup plus étendu.Le nombre de cas d'utilisation dans l'industrie et la vie quotidienne est presque infini. Des sec-

teurs Bancaire, Dossiers fiscaux, Assurance, Enregistrement de biens immobiliers et fonciers, Chaîne d'approvisionnement, Sciences de la vie et soins de santé ... peuvent tous bénéficier des contrats intelligents visant à éliminer les tiers.

C'est précisément dans cette optique que s'insère ce travail. Aujourd'hui, de nombreux passionnés travaillent sur des projets visant à résoudre les problèmes à l'aide de la technologie de la blockchain et des contrats intelligents. Ce travail de recherche permet de mieux comprendre cet environnement novateur et d'appréhender le vaste éventail des possibilités qu'offrent les Smart Contrats. Plus encore, à partir de la mise en pratique de cette théorie à travers un projet concret **iCubeCoin** (Projet visant à mettre en place une cryptomonnaie iCubeCoin), ce travail permettra ainsi de se forger une connaissance solide de l'environnement de la cryptomonnaie et apprécier par lui-même toute la force de ce mouvement.

Ce travail a été réalisé dans le cadre d'un stage de Validation de Master Informatique, option Système Intelligent et Multimédia. J'ai donc travaillé durant 6 mois à la DISFE, la structure qui m'a accueillie dans ce cadre. J'ai pu assister à des réunions, meetups sur le sujet de la "Blockchain". J'ai également eu le privilège de présenter le projet iCubeCoin à la 6^{ème} édition de la journée de l'innovation du Groupe la poste (Journée de partage sur les projets et les innovations en cours).

Chapitre 2

Présentation de la structure d'accueil

J'ai été accueilli dans le cadre de mon stage par la DISFE (le pôle de La Banque Postale et du Réseau La Poste qui sont tous les deux des branches du Groupe La Poste).

2.1 Le Groupe La Poste

Le Groupe La Poste est, après l'État, l'un des principaux employeurs de France. Il contribue de longue date au bien-être social et économique notamment en France. Le Groupe La Poste a développé avec les Français une véritable proximité en facilitant leur quotidien et en s'inscrivant durablement dans leurs territoires. Il réunit près de 260000 postiers entrant chaque jour en relation avec 65 millions de personnes, partout en France, et animés par des valeurs citoyennes qui sont depuis toujours au cœur de l'identité postale. Fort de sa présence territoriale et de ces valeurs, socles de la confiance des Français dans La Poste, le Groupe assure quatre missions de service public, pleinement intégrées à ses activités :

- la distribution du courrier 6 jours sur 7 au domicile de tous les Français,
- la contribution à l'aménagement du territoire,
- le transport et distribution de la presse,
- l'accessibilité bancaire

Le Groupe La Poste affirme sa position d'opérateur de services de proximité humaine. Outre ses 17 200 points de contacts en France, Le Groupe La Poste est aujourd'hui présent dans 44 pays et sur 4 continents. Il s'orchestre autour de 5 branches-métiers majeures permettant de répondre aux enjeux d'aujourd'hui et de demain. Les 5 branches métiers sont représentées dans l'image ci-dessus tirée du [Livret d'accueil - Direction Contrôle de Gestion](#) [1].



FIGURE 2.1 – Les 5 branches métiers du Groupe La Poste

Service courrier-colis : Il s'agit des activités historiques du groupe, ce pourquoi il est le plus connu. On y retrouve donc les services de courrier et colis de la maison mère ainsi que ses filiales et coentreprises comme Mediaposte, Viapost Services, Asendia ou encore SOGEC. Cela représente 55,5% des effectifs du groupe.

GeoPost : Cet axe représente le transport de colis express dans le monde entier avec, notamment le célèbre Chronopost. Mais également d'autres services comme DPD, Seur, Interlink Express et Pickup Services. On y retrouve 13% des effectifs dans cette branche.

Le réseau La Poste : Il s'agit là d'un réseau de proximité (développement des bureaux de poste du territoire). On y retrouve également la Poste Mobile, une filiale du groupe en partenariat avec SFR. 21,5% des effectifs travaillent dans ce réseau.

Le Numérique : La branche pilote l'expérience client en ligne et la transformation du Groupe. Elle permet la création de nouvelles offres et assure le développement numérique du Groupe. On y retrouve par exemple, Decapost, Start'inPost et Mediapost Communication. La branche Numérique représente 2,5% des effectifs du groupe.

La Banque postale : La branche de la banque postale regroupe les activités bancaires de la Poste. On y retrouve une banque de détail, de la gestion d'actifs et des services

d'assurances. En 2014, elle compte 10,7 millions de clients. Cela représente 7,5% des effectifs du groupe.

Ma French Bank : Nouvelle branche métier intégrant la nouvelle banque complètement mobile du groupe la Poste.

2.2 La DISFE

2.2.1 Présentation

La DISFE est la **Direction de l'Informatique des Services Financiers et de l'Enseigne La Poste**. Elle assure la maîtrise d'œuvre informatique des systèmes d'information de La **Banque Postale** et du **Réseau La Poste**, avec pour objectifs de délivrer la meilleure qualité de service aux clients, aux utilisateurs, et d'optimiser les coûts. Elle fait partie du **Pôle solutions** qui rassemble également sous une même responsabilité la Direction des paiements (DP) et la Direction performance et changement (DPC),

Énoncé sur son portail informatique intranet, La DISFE accompagne les enjeux stratégiques de la Banque et du Réseau en assurant cinq missions principales :

- **conduire la politique d'investissement en matière de système d'information en conjuguant productivité, réactivité et rentabilité.** Il s'agit d'adapter les systèmes d'informations selon les besoins des utilisateurs, pour assurer la réalisation et l'intégration des projets de la Banque et du Réseau en garantissant la cohérence et l'exploitabilité des SI,
- **mettre à disposition de l'ensemble des utilisateurs des services optimisés et conformes à leurs attentes.** Objectif qualité de service afin d'améliorer la satisfaction des utilisateurs internes (les collaborateurs) et externes (le grand public),
- **maîtriser les risques d'altération du service rendu dans le respect des contraintes réglementaires et économiques,**
- **rationaliser l'activité informatique et innover en introduisant une analyse permanente de la valeur.** Parce que l'innovation n'est pas une fin en soi, la capacité de mutualisation et de simplification des systèmes d'informations contribue au développement commercial du Groupe,
- **maîtriser le développement professionnel de chaque collaborateur dans le respect des besoins de l'entreprise et des valeurs du Groupe.**

2.2.2 Organisation

La DISFE est structurée par directions études et développements, directions techniques et de production, et fonctions de gestion. Cette organisation comprend :

- Direction des SI filiales et conduite du changement métiers SI

- Direction des ressources humaines
- Direction de la maîtrise des risques
- Coordination des programmes distribution
- Coordination des programmes core banking
- Programme Ma French Bank
- Coordination technique et urbanisme
- Direction pilotage et moyens généraux
- Direction de la production informatique
- Direction de l'architecture et des infrastructures
- Coordination des programmes risques et finance
- Coordination des programmes personnes morales
- Patrimoine applicatif de la Banque
- Direction du système d'information du Réseau

Dans cette organisation, j'ai fait parti de la **Direction de l'architecture et des infrastructures (DAI)** ayant pour mission de :

- concevoir, entretenir et supporter les architectures et les services : de l'environnement de travail utilisateur, des infrastructures en Datacenter & Hors-Datacenter, du cadre de développement et d'exécution des applications,
- accompagner les équipes projets et programmes dans la mise en œuvre des architectures,
- assurer la gouvernance des choix et des évolutions d'architecture,
- piloter les coûts télécoms,
- animer l'innovation et la veille technologique,
- piloter les projets d'évolution des capacités IT.

Chapitre 3

Contexte d'étude et problématique

3.1 Projet iCubeCoin (I3C)

La Banque Postale mène des actions solidaires en cohérence avec les valeurs postales de proximité et de service au plus grand nombre qu'elle porte au sein de la société. Sa démarche s'inscrit pleinement dans celle du Groupe La Poste et repose essentiellement sur l'engagement citoyen de ses collaborateurs sur tout le territoire. En complément de sa démarche de mécénat (mise à disposition de l'ensemble de ses supports de communication pour soutenir de grandes causes notamment avec les appels à dons), La Banque Postale encourage et soutient ses collaborateurs dans leur engagement citoyen. Parce qu'ils sont les premiers acteurs au quotidien du développement de l'entreprise, la Banque encourage ses collaborateurs à jouer un rôle à part entière dans sa démarche sociétale. Afin d'encourager l'investissement citoyen des collaborateurs et valoriser les idées neuves, responsables, solidaires et écologiques il est proposé la **création d'une monnaie virtuelle** utilisable à travers une application spécifique permettant la réception, le transfert et l'échange de cette monnaie à travers un kiosque de services et produits.

3.2 Objectif

La reconnaissance est un élément clé du bien-être en entreprise. Cette reconnaissance peut être portée par la création d'une monnaie virtuelle, le **iCubeCoin** utilisable à travers une application proposant plusieurs fonctionnalités. Comme toutes les entreprises, la DISFE cherche à catalyser la collaboration en son sein. De plus, elle souhaite renforcer son unité en tant que groupe malgré des entités distinctes et développer une horizontalité bien loin des hiérarchies verticales classiques. Une monnaie interne, commune à tous les collaborateurs, peu importe leur localisation ou leur rôle hiérarchique, permet de trouver un point commun entre tous les acteurs de l'entreprise. C'est aussi une manière de permettre à tous de mieux échanger, y compris entre

équipes distinctes, affirmant ainsi une hiérarchie horizontale. Ce projet a pour objectif de :

1. Favoriser la création de valeur en récompensant l'investissement individuel ou collectif
2. Catalyser la collaboration et renforcer l'unité du groupe (créer des liens entre des équipes distantes géographiquement, appartenant à différentes entités et développer une horizontalité bien loin des hiérarchies verticales classiques)
3. Améliorer la qualité de vie au travail, important gisement de compétitivité
4. Obtenir une cartographie de l'ensemble des actions locales et nationales

3.3 Identification des besoins

1. Créer son portefeuille et recevoir des i3 coins

Tout collaborateur pourra créer son compte i3 à travers l'application dédiée. Et gagner des i3coins selon les chemins suivants :

- (a)
 - i. feu vert donné par l'entreprise (Direction/RH/Com/IET etc.) pour la mise en œuvre d'une activité
 - ii. campagne de recrutement (ex. Envol, appel à bénévolat, etc.)
 - iii. le recrutement enclenche le paiement de la somme de i3C prévue
- (b)
 - i. l'initiative est portée par une "communauté" des collaborateurs bénévoles (ex. AGAPE= association du personnel, I3network, etc.)
 - ii. l'entreprise fait un versement à ceux qui portent l'initiative (conditions à définir)
 - iii. les porteurs du budget récompensent l'investissement

2. Transférer des i3coins d'un portefeuille vers un autre

3. Possibilité d'effectuer des virements (ex. : transférer des i3C aux collègues qui gèrent les paniers bio, au collègue qui a préparé une présentation, etc.).

4. Gérer des « budgets » i3coins dans le cadre d'un projet (fonctionnalité liée à des profils particuliers)

Les collaborateurs identifiés comme étant porteurs d'un projet spécifique bénéficient d'un budget spécifique permettant de récompenser les participants.

5. Pouvoir échanger ses i3 coins pour des avantages non monétaires (Kiosque de services et produits)

A monter avec les directions compétentes (RH, etc.).

Exemples possibles : un nombre de Cafés aux machines prévues dans les espaces détente, temps disponible pour des projets utiles à l'entreprise, invitation au restaurant pour les collègues de son service, formations spécifiques ou complémentaires, participation à des forums et des événements, services de conciergerie, services d'aide à la personne, possibilité de transformer les i3coins en don pour des associations caritatives partenaires de la Banque Postale (LBP), etc.

6. Monnaie « vivante » afin d'encourager l'utilisation des i3coins
Lorsqu'un collaborateur ne dépense pas de l'argent au bout d'un certain temps (à définir), il perd un pourcentage de la somme gagnée. La somme prélevée sera ainsi réinjectée dans le système global (à définir - ex. possible : don pour une des associations partenaires de LBP
7. Pouvoir proposer des nouveaux produits et services afin d'élargir l'offre du Kiosque
Le kiosque doit pouvoir s'adapter aux besoins des collaborateurs et donner l'envie de s'investir.
8. Pouvoir proposer des nouveaux projets afin d'enrichir le Référentiel des actions éligibles à la distribution des i3 coins
Créer et faire vivre le référentiel d'initiatives.
9. Reporting permet de suivre en temps réel l'utilisation de la monnaie (identifier des tendances dans l'entreprise)
10. Introduire un volet gamification (ex. : e-sport, organisation des challenges, etc.) pouvant inciter à plus d'utilisation

3.4 Domaine d'étude

Ce projet apporte une réflexion nouvelle sur le monde de la **Blockchain**. L'idée est de créer une monnaie virtuelle pour encourager certaines initiatives des collaborateurs de La Banque Postale et valoriser tout type d'investissement. Le but est d'expérimenter une nouvelle technologie (la Blockchain) dont les principaux avantages sont : désintermédiation, traçabilité, consensus distribué. C'est justement sur cet environnement de confiance que soulève la création d'une plateforme décentralisée d'échange de jetons I3C. Grâce à cette plateforme, la communauté impliquée se rassemble et les gens se réunissent.

3.5 Spécification des technologies

3.5.1 Qu'est-ce qu'une Blockchain ?

3.5.1.1 Définition générale

La blockchain est une technologie de stockage et de transmission d'informations, transparente, sécurisée, et fonctionnant sans organe central de contrôle. [définition de Blockchain France\[2\]](#)

Une blockchain est un registre distribué qui est géré entre pairs et fonctionnant sans l'intervention d'une autorité centrale, garantissant ainsi la transparence et la sécurité du système. La bonne analogie pour aider à comprendre le fonctionnement de la blockchain est d'y voir un grand registre ou un grand livre où tout est noté jusqu'au moindre détail. Il est donc possible de garder une traçabilité de tous les échanges, et de

s'assurer de l'état des comptes, de la même manière que le ferait un audit comptable. La seule différence est qu'une blockchain n'est pas un grand livre, mais autant de registres que de participants, assurant tous la sécurité et la disponibilité de l'information.

Par extension, une blockchain est donc une base de données contenant tout l'historique des échanges réalisés depuis sa création, et partagée par ses différents utilisateurs. Chaque participant du réseau est un "nœud".

Il existe des blockchains **publiques**, ouvertes à tous, et des blockchains **privées**, dont l'accès et l'utilisation sont limitées à un certain nombre d'acteurs. La première donne la possibilité de consulter l'intégralité de l'information. La seconde restreint la lecture complète de certaines informations (plus de confidentialité, mais moins de transparence et donc de sécurité), ou limite le nombre d'acteurs qui possèdent les droits nécessaires.

3.5.1.2 Caractéristiques essentielles d'une Blockchain

4 caractéristiques distinguent la blockchain :

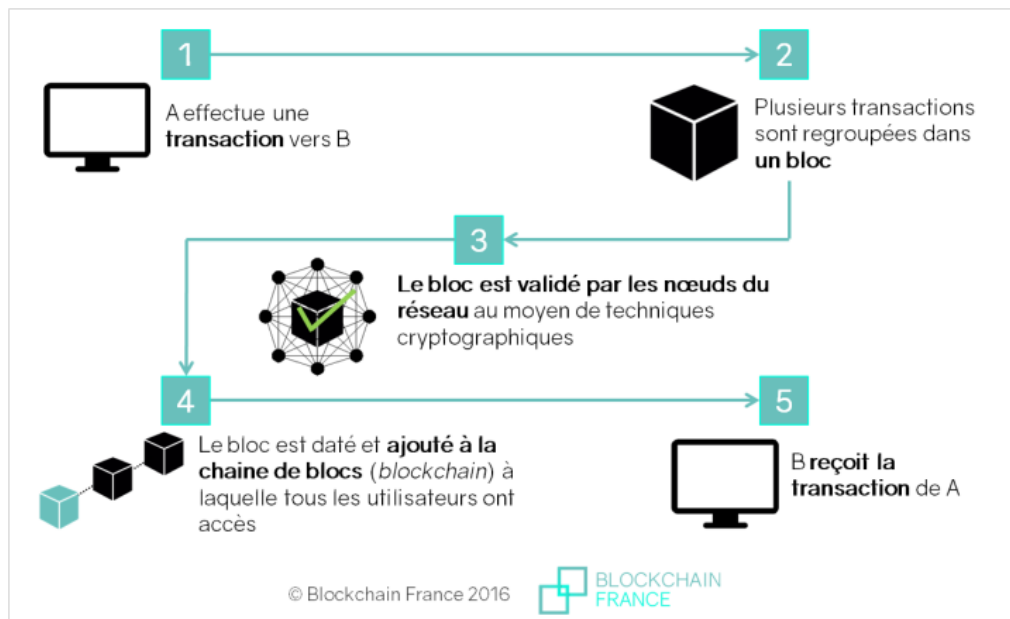
- Il est conçu pour être distribué et synchronisé sur des réseaux, ce qui le rend idéal pour les réseaux d'entreprise multi-organisationnels tels que les chaînes d'approvisionnement ou les consortiums financiers. Il encourage également les organisations à sortir de derrière leur pare-feu et à partager des données.
- Vous ne pouvez pas simplement faire ce que vous voulez des données. Les types de transactions que l'on peut effectuer sont convenus à l'avance entre les participants et stockés dans la blockchain sous la forme de « contrats intelligents », ce qui permet de s'assurer que chacun respecte les règles.
- Avant de pouvoir exécuter une transaction, toutes les parties concernées doivent se mettre d'accord sur le fait que la transaction est valide. Par exemple, si vous enregistrez la vente d'une vache, cette vache doit vous appartenir ou vous ne recevrez pas d'accord. Ce processus est appelé « consensus » et permet de conserver les transactions inexactes ou potentiellement frauduleuses en dehors de la base de données.
- Immutabilité des données : une fois que vous avez convenu d'une transaction et que vous l'avez enregistrée, celle-ci ne peut jamais être modifiée. Vous pouvez ensuite enregistrer une autre transaction sur cet actif pour changer son état, mais vous ne pouvez jamais masquer la transaction d'origine. Cela donne l'idée de la provenance des actifs, ce qui signifie que pour tout actif, vous pouvez indiquer où il se trouve, où il se trouve et ce qui s'est passé tout au long de sa vie.

[de IBM\[3\]](#)

3.5.1.3 Fonctionnement d'une Blockchain

Les transactions effectuées entre les utilisateurs du réseau sont regroupées par blocs. Chaque bloc est validé par les nœuds du réseau appelés les "mineurs", selon des techniques qui dépendent du type de blockchain. Une fois le bloc validé, il est horodaté et

ajouté à la chaîne de blocs. La transaction est alors visible pour le récepteur ainsi que l'ensemble du réseau.” [Blockchain France, 2016](#)[4]



Lorsqu'une transaction est effectuée, elle est regroupée avec d'autres au sein d'un bloc et ne peut plus être modifiée. Les mineurs valident le bloc grâce à des techniques cryptographiques. Une fois le bloc validé, il est ajouté à la chaîne de blocs accessible à tous les utilisateurs. Rien ne peut être modifié ni effacé : il faudrait ajouter une nouvelle transaction en cas d'erreur. L'information est donc très difficilement falsifiable.

Néanmoins tout le monde peut accéder à une blockchain car il n'y a pas de contrôle d'identité des utilisateurs. Il est difficile de savoir si l'utilisateur est une personne physique ou morale. Chaque nœud pourrait donc inscrire une transaction sans connaître la fiabilité de l'opérateur.

L'incapacité de pouvoir faire confiance aux membres du réseau fait place à un "consensus". En effet, pour que le "bloc" soit ajouté à la "chaîne", il doit être validé par tous les nœuds du réseau.

3.5.2 Qu'est-ce qu'une crypto-monnaie?

Définition Une crypto-monnaie est un actif ou une forme numérique de monnaie échangeable, construit sur une technologie blockchain qui n'existe qu'en ligne. Il n'y a pas d'option permettant d'obtenir une crypto-monnaie sous forme de papier ou de monnaie. Les crypto-monnaies utilisent la cryptographie pour vérifier et sécuriser les transactions, d'où leur nom. Il existe actuellement plus de 2000 crypto-monnaies différentes dans le monde et de nombreuses personnes les considèrent comme le pivot d'une économie future plus juste.

Dans sa forme la plus simple, une crypto-monnaie fonctionne en enregistrant les transactions dans une base de données pour déterminer la quantité de cette devise que chaque individu, ou son adresse, détient. En ce sens, le système n'est pas si différent du fonctionnement actuel des banques. Par exemple, l'argent que vous dépensez en ligne suit des principes similaires : vous envoyez de l'argent de votre compte bancaire à un autre compte en déduisant du montant numérique que vous avez associé à votre compte, c'est-à-dire votre solde. Ce n'est rien de plus que des informations enregistrées dans une base de données, aucun échange physique n'a lieu.

La différence entre une crypto-monnaie et un jeton Il est important de garder à l'esprit qu'il existe 2 types de crypto-monnaies : l'une est une crypto-monnaie de monnaie et l'autre est un jeton. La principale différence entre eux est que les crypto-monnaies ont leur propre blockchain, alors que les jetons utilisent la blockchain ou un autre réseau principal déjà construit.

Portefeuille un portefeuille crypto-monnaie est essentiellement un fichier composé de deux clés privée et publique. Votre clé publique est votre adresse de crypto-monnaie, , tandis que la clé privée vous permet de dépenser de l'argent depuis ce compte. Il est important de conserver une sauvegarde sécurisée de votre portefeuille, même si vous utilisez un fournisseur de portefeuille de crypto-monnaie en ligne, car perdre votre portefeuille, c'est-à-dire le supprimer définitivement, vous empêchera d'accéder à vos fonds.

3.5.3 Mécanismes de consensus

Afin de valider si l'ajout de nouvelles informations à la chaîne de blocs (par exemple un enregistrement de transaction) est légitime, les nœuds doivent parvenir à un accord. Cet accord est un "consensus". Le consensus est un mécanisme spécifique pré-défini qui garantit un séquençement correct des transactions sur la chaîne de blocs. Dans le cas des crypto-monnaies, un tel séquençement est nécessaire pour résoudre le problème de «Double dépense» (c'est-à-dire le fait qu'un seul et même instrument de paiement ou bien peut être transféré plus d'une fois si les transferts ne sont pas enregistrés et contrôlés de manière centralisée).

Un mécanisme de consensus peut être structuré de plusieurs manières. Ci-après, les deux exemples de mécanismes consensuels les plus connus et les plus couramment utilisés sont également abordés : la preuve de travail "proof-of-work" (PoW) et la preuve de participation "proof-of-stake" (PoS).

3.5.3.1 Proof of Work (PoW)

Proof Of Work (POW) est le premier mécanisme de consensus blockchain et a été utilisé pour la première fois par Bitcoin. De nombreuses crypto-monnaies ont suivi l'exemple de Bitcoin et ont également adopté ce mécanisme de consensus.

Le processus de validation du travail est appelé extraction et les nœuds sont appelés mineurs. Les mineurs résolvent des énigmes mathématiques complexes qui nécessitent beaucoup de puissance de calcul. Le premier à résoudre le puzzle crée un bloc et reçoit une récompense pour la création d'un bloc. Ces énigmes mathématiques ont des propriétés intéressantes. Tout d'abord, ils sont asymétriques, ce qui signifie qu'il faut beaucoup de temps pour trouver la réponse, mais qu'il est facile de vérifier si une réponse est correcte.

Deuxièmement, le seul moyen de résoudre ces énigmes est de «deviner» la réponse. Il n'est pas possible de résoudre les énigmes plus rapidement en utilisant une autre méthode que les essais et les erreurs. Cela signifie également que si l'on veut trouver la solution au puzzle plus rapidement, il faudrait plus de puissance de calcul, ce qui peut coûter très cher. Enfin, la difficulté de ces énigmes change en fonction de la vitesse d'extraction des blocs. Pour maintenir un approvisionnement constant en nouvelles crypto-monnaies, des blocs doivent être créés dans un certain délai. Si les blocs sont créés trop rapidement, les énigmes deviennent plus difficiles et, si elles sont créées trop lentement, les énigmes deviennent plus faciles.

3.5.3.2 Proof of Stake (PoS)

PoS se fonde sur le principe selon lequel ceux qui possèdent la plupart des crypto-monnaies d'un réseau ont tout intérêt à ce que le réseau soit maintenu et que la valeur de ses crypto-monnaies soit élevée.

Dans un système qui utilise la PoS (appelé Preuve de participation), un processus aléatoire est utilisé pour déterminer qui doit produire le bloc suivant. Les utilisateurs peuvent jouer leurs crypto-monnaies pour devenir un validateur, ce qui signifie qu'ils verrouillent leurs crypto-monnaies pendant un certain temps. Après quoi, ils sont éligibles pour produire des blocs.

Les validateurs sont également récompensés pour leur travail. La récompense que le validateur reçoit pour la création du prochain bloc dépend encore une fois de la conception de la blockchain. Habituellement, ils reçoivent tout ou partie de tous les frais de transaction de toutes les transactions du bloc qu'ils ont créé ou reçoivent un montant fixe de crypto-monnaies (généralisé par l'inflation).

PoS est non seulement beaucoup plus économe en énergie que le système PoW, mais présente une autre distinction majeure. Dans un système de preuve de travail, un mineur ne peut posséder aucune des crypto-monnaies qu'il exploite, ce qui signifie qu'il cherche uniquement à maximiser ses profits sans améliorer réellement le réseau. Dans un système de preuve de participation, les validateurs sont beaucoup plus incités à entretenir réellement le réseau car ils détiennent les crypto-monnaies de la blockchain sur laquelle ils sont en train de valider.

3.5.4 Qu'est-ce qu'Ethereum?

Définition Ethereum est une plate-forme ouverte qui permet aux développeurs de créer et de déployer des applications décentralisées telles que des contrats intelligents et d'autres applications juridiques et financières complexes. Ethereum fonctionne avec une crypto-monnaie l'Ether.

Ethereum est un Bitcoin programmable dans lequel les développeurs peuvent utiliser la blockchain sous-jacente pour créer des marchés, des grands livres, des organisations numériques et d'autres possibilités sans fin qui nécessitent des données et des accords immuables, le tout sans intermédiaire. Lancé en 2015, Ethereum a été conçu par le prodigieux Vitalik Buterin, qui a compris les utilisations potentielles de la technologie de blockchain sous-jacente de Bitcoin

3.5.5 Qu'est ce qu'un Smart Contract?

Définition Les contrats intelligents sont des contrats préprogrammés avec un ensemble de règles et réglementations définitives qui s'appliquent automatiquement, sans intermédiaire.

Fonctionnement Les contrats intelligents fonctionnent en suivant des instructions simples «si / quand... alors...» qui sont écrites dans le code d'une blockchain. Un réseau d'ordinateurs exécute les actions (déblocant des fonds vers les parties appropriées; immatriculation d'un véhicule; envoyant des notifications; émettant un ticket) lorsque des conditions prédéterminées ont été remplies et vérifiées. La blockchain est ensuite mise à jour une fois la transaction terminée.

Avantages Les avantages des contrats intelligents vont de pair avec la blockchain.

- **Rapidité et précision :** les contrats intelligents sont numériques et automatisés. Vous ne perdez donc pas de temps à traiter les écritures, à réconcilier et à corriger les erreurs souvent écrites dans des documents remplis manuellement. Le code informatique est également plus précis que le jargon juridique dans lequel sont écrits les contrats traditionnels.
- **Confiance :** les contrats intelligents exécutent automatiquement des transactions selon des règles prédéterminées, et les enregistrements signés de ces transactions sont partagés entre les participants. Ainsi, personne ne doit se demander si l'information a été modifiée pour son bénéfice personnel.
- **Sécurité :** les enregistrements de transaction Blockchain sont signés, ce qui les rend très difficiles à pirater. Étant donné que chaque enregistrement individuel est connecté aux enregistrements précédents et suivants sur un grand livre distribué, il serait nécessaire de modifier toute la chaîne pour modifier un seul enregistrement.

- **Économies** : les contrats intelligents suppriment le besoin d'intermédiaires, car les participants peuvent faire confiance aux données visibles et à la technologie pour exécuter correctement la transaction. Il n'est pas nécessaire de faire appel à une personne supplémentaire pour valider et vérifier les termes d'un accord car celui-ci est intégré au code.

3.6 Problématique

Comment permettre aux collaborateurs, qui n'ont aucune façon d'avoir confiance l'un envers l'autre, de gagner, d'échanger et de dépenser cette monnaie iCubeCoin en toute sécurité? Comment garantir la transparence afin que cette monnaie soit acceptée de tous?

3.7 Résultat attendu

Ma mission est de créer la crypto-monnaie i3CubeCoin et de développer une application qui servira d'interface pour gérer cette crypto-monnaie en tenant compte de la logique métier détaillée dans les spécifications du projet.

État de l'art

4.1 L'histoire des cryptomonnaies

La plupart des personnes ont entendus parler des crypto-monnaies avec l'avènement de la première crypto-monnaie créée, Bitcoin. Cependant, lorsque Satoshi Nakamoto, le mystérieux fondateur de Bitcoins, a créé la première crypto-monnaie viable au monde, il ne cherchait pas du tout à inventer une monnaie.

La monnaie numérique est un concept qui existait bien avant Bitcoin. Le meilleur exemple est une société appelée DigiCash Inc., fondée en 1989, qui tente de créer la première monnaie numérique largement utilisée au monde. DigiCash était une société de monnaie électronique créant un protocole de paiement anonyme basé sur la cryptographie. Toutefois, après avoir échoué à obtenir sa convertibilité avec les fiat (monnaie fiduciaire émise par les banques centrales, qui repose sur la confiance que l'on a en ces institutions émettrices. Exemple : \$, €, £...), entre autres problèmes, DigiCash a été contraint de se déclarer en faillite en 1998.

Dix ans plus tard, peut-être en réaction à la crise économique de 2008, un développeur inconnu, connu seulement à ce jour comme Satoshi Nakamoto, a publié un [livre blanc](#) [20] d'un système décentralisé, électronique peer-to-peer cash, qui est devenu Open Source (ce qui signifie que tout développeur peut y contribuer) en 2009. Ce projet, appelé Bitcoin, était le premier exemple de crypto-monnaie fonctionnelle.

Depuis lors, des milliers de crypto-monnaies et de jetons utilitaires supplémentaires ont vu le jour, allant de projets sérieux visant à changer le monde en permettant l'adoption de la technologie de blockchain.

4.2 Les cryptomonnaies sur le marché






4.2.1 Classification des cryptomonnaies sur le marché

Après avoir connu une croissance soutenue au cours des deux dernières années, le marché des crypto-monnaies a explosé en 2017, enregistrant une hausse de plus de

1,200%. À l'heure actuelle, plusieurs centaines de cryptomonnaies sont en circulation (avec une capitalisation boursière totale bien supérieure à 300 milliards d'euros), et d'autres continuent à apparaître régulièrement. Afin de bien saisir ce marché émergent et de mener une étude sérieuse, nous avons choisi d'abord d'analyser les propriétés clés du bitcoin, la crypto-monnaie la plus connue, puis d'aborder les caractéristiques principales d'un certain nombre de crypto-monnaies alternatives, mieux connues sous le nom de "Altcoins". Les altcoins sont toutes des cryptomonnaies qui constituent une alternative à Bitcoin. En résumé, il existe deux types d'altcoins :

- Les altcoins construites à l'aide du protocole open source original de Bitcoin, avec un certain nombre de modifications apportées à ses codes sous-jacents, en concevant une nouvelle cryptomonnaie avec un ensemble de fonctionnalités différent. Un exemple d'un tel Altcoin est Litecoin.
- Les altcoins qui ne sont pas basés sur le protocole open source de Bitcoin, mais qui ont leur propre protocole et grand livre distribué. Ethereum et Ripple sont des exemples bien connus de ces altcoins.

Cette étude se concentrera sur les Altcoins qui font partie des tops 5 des monnaies virtuelles ayant la plus grosse capitalisation boursière. [Coinmarketcap.com](https://coinmarketcap.com) a réalisé le classement des crypto-monnaies en fonction de leur valorisation boursière.

Crypto-monnaies ▾		Plateformes d'échange ▾	Liste de suivi			
#	Nom	Cap. Marché	Prix	Volume (24h)	Offre en Circulation	
1	 Bitcoin	\$208 432 613 207	\$11 702,05	\$33 675 301 921	17 811 625 BTC	
2	 Ethereum	\$29 050 059 485	\$271,86	\$11 835 517 367	106 855 033 ETH	
3	 XRP	\$14 254 759 215	\$0,334881	\$2 669 534 281	42 566 596 173 XRP *	
4	 Litecoin	\$6 407 133 730	\$102,33	\$5 215 894 312	62 614 987 LTC	
5	 Bitcoin Cash	\$6 202 250 595	\$346,77	\$2 504 948 732	17 885 575 BCH	

4.2.2 Bitcoin (BTC)

Le [Bitcoin](#) [5] est la star des cryptomonnaies, la première à avoir attiré l'attention du grand public. Créée en 2008, elle a fêté ses dix ans en 2018. Son inventeur, Satoshi Nakamoto, reste mystérieux. On ne sait pas s'il s'agit d'une personne ou d'un groupe de personnes. Cette inconnue correspond à la philosophie qui sous-tend le Bitcoin, à savoir fournir une monnaie totalement décentralisée, sans contrôle des États, d'une entreprise ou d'une personne.

Le Bitcoin, comme les autres cryptomonnaies, repose sur la technologie blockchain. Il s'agit d'une technologie de stockage et de transmission d'informations, fonctionnant sans organe central de contrôle. Lorsque deux utilisateurs échangent de la monnaie cryptée, ils sont les deux seuls à avoir accès à la transaction, ce qui représente un avantage certain en termes de sécurité.

4.2.3 Ethereum (ETH)

L' [Ethereum](#)[6] lié à sa crypto-monnaie nommée « Ether » (ETH), est un système créé par Vitalik Buterin et lancé en juillet 2015. Alors que le système Bitcoin ne sert qu'à supporter sa monnaie, aussi nommée Bitcoin, et à proposer un système de paiement décentralisé, le système Ethereum va plus loin avec la mise en place de « smart contracts », ou « contrats intelligents ». La force de l'Ethereum, par ailleurs comparable au Bitcoin, repose sur sa blockchain, la technologie sur laquelle il est basé. Celle-ci peut servir de base à une multiplicité d'applications, contrairement à celle du Bitcoin destinée uniquement aux transactions. De plus, les "environnements clients" pour Ethereum reposent sur les langages les plus courants comme C++ (Cpp-ethereum), Haskell (ethereumH), JavaScript (EthereumJS-lib), Python (Pyethapp) ou encore le langage Go (Go-ethereum ou Geth)

4.2.4 Ripple (XRP)

[Ripple](#) [7] a connu une forte croissance en 2017. Le Ripple a la spécificité d'avoir le soutien des banques : UBS, UniCredit ou encore Santander utilisent déjà sa blockchain, qui permet de transférer des fonds sans frais.

Rejeté par les «geeks» et les «cypherpunks» (car les banques et institutions financières y voyaient là un développement de leur activité avec des solutions de paiements numériques), qui défendent les libertés individuelles et les monnaies cryptées comme solution pour sortir de l'ornière des institutions financières et des États, le Ripple semble promis à un bel avenir grâce au soutien des banques.

4.2.5 Bitcoin Cash (BCH)

Surnommé «le vrai Bitcoin» par ses adeptes, le [Bitcoin Cash](#)[8] est né le 1er août 2017 d'un schisme («hard fork») avec le Bitcoin. S'il est basé sur la même blockchain, il est plus rapide et garantit des transactions moins onéreuse que le Bitcoin. Mais ce dernier devrait connaître prochainement des évolutions lui permettant de rattraper son déficit

4.2.6 Litecoin (LTC)

Basé sur la technologie Bitcoin, le [Litecoin](#) [9] est l'une des cryptomonnaies les plus connues. Créée en 2011 par Charles Lee, un ancien employé de Google, qui est parti

du code source de Bitcoin (en open source), pour mettre au point son propre code, le Litecoin est une cryptomonnaie distribuée sous licence libre.

La blockchain sur lequel est basée le Litecoin a ainsi été amélioré par rapport à celui du Bitcoin. Cela a notamment permis d'accélérer le processus de vérification et donc d'augmenter la rapidité des transactions, supérieure actuellement à celle de ses concurrents. Et les frais de transaction sont bien plus faibles que ceux du Bitcoin. Le Litecoin est ainsi particulièrement destiné aux transactions quotidiennes.

4.3 4 méthodes différentes pour créer sa propre crypto-monnaie

4 solutions s'offrent à nous pour développer et lancer sa propre devise : Créer sa propre blockchain (partir de zéro), Forker une crypto-monnaie existante (dupliquer un code) , Créer un jeton Ethereum ou Utiliser un service de création de crypto-monnaie (passer par des sociétés).

4.3.1 Créer sa propre blockchain

Pour ceux qui possèdent une connaissance approfondie du code et savent comment le manipuler, il est possible de créer sa cryptomonnaie à partir de zéro. Il faut donc rédiger son propre code ligne par ligne pour créer ses propres jetons. Un serveur personnalisé, un espace de stockage conséquent et un ordinateur puissant sont également nécessaires pour se lancer dans cette opération.

C'est un processus assez long et complexe qui n'est pas à la portée de tous. Pour les geeks avisés et les férus de technologie, il existe de nombreux tutoriels sur le web,

Mais avant de suivre pas-à-pas les indications des professionnels, il est important de garder à l'esprit que la principale qualité d'une cryptomonnaie est sa sécurité. Pour que votre devise soit inviolable, il faut que votre code soit particulièrement bien écrit et cela demande d'avoir un niveau d'expert en sécurité informatique.

4.3.2 Forker une crypto-monnaie existante

Des blockchain open source ont été développées et peuvent être réutilisées facilement pour créer une nouvelle crypto-monnaie.

Le code open source d'une blockchain peut être repris et modifié pour développer une nouvelle devise. Appelé Fork, on parle de hard fork lorsque la blockchain subit d'importantes modifications et de soft fork quand la blockchain est légèrement modifiée.

On peut par exemple prendre l'exemple du bitcoin, la devise la plus 'forkée' car la plus populaire. Il existe des dizaines de forks du bitcoin et les plus connus sont sans doute le litecoin et le bitcoin cash.

Les codes des crypto-monnaies open source sont pour la plupart disponibles sur le site Github. Bitcoin, par exemple, est un logiciel open source que vous pouvez trouver sur [GitHub](#) [10] :

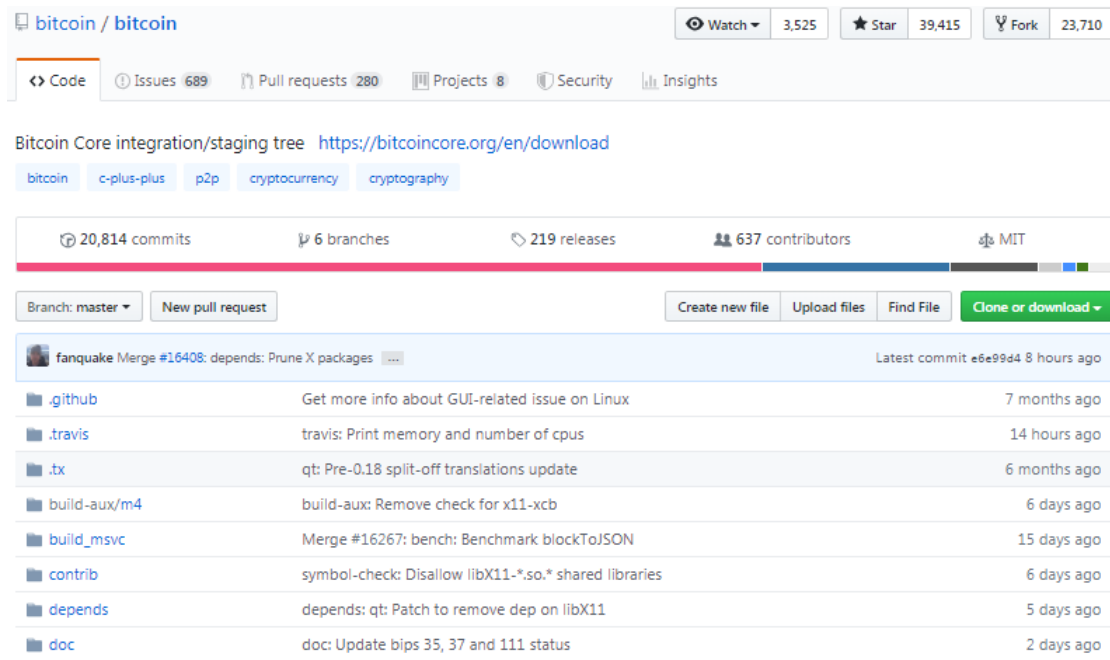


FIGURE 4.1 – Code source Bitcoin sur Github

4.3.3 Créer un jeton

La fondation Ethereum met à disposition de tous un code open-source téléchargeable sur son site. Ce code permet de créer des jetons ERC-20. Ces jetons contiennent un certain nombre d'informations concernant le transfert de jetons ainsi que des méta-données comme le nombre de jetons en circulation, ou le solde du compte contenant les jetons.

Les jetons numériques (tokens en anglais) peuvent représenter n'importe quel bien échangeable : cryptomonnaies de monnaie, points de fidélité, certificats, reconnaissances de dette, objets de jeu, etc. Dans le cas présent, ces jetons serviront d'unités de base à notre monnaie. Les tokens ERC-20 étant créés avec les standards d'Ethereum, la monnaie qui verra le jour sera instantanément compatible avec les portefeuilles Ethereum et tout autre client ou contrat qui utilise les mêmes standards.

Le code de ces jetons va servir de base à la création d'une nouvelle monnaie cryptée. Essentiellement, les développeurs n'ont qu'à changer quelques paramètres dans les lignes de code de l'ERC-20, avec un éditeur de texte de programmation comme Sublime Text, pour avoir les fondations de la nouvelle monnaie.

4.3.4 Utiliser un service de création de crypto-monnaie

Aujourd'hui, plusieurs plateformes se proposent de créer des monnaies cryptées en quelques jours. Pour une certaine somme, des sociétés comme CryptoLife, Wallet Builders ou Coin Creator créent des monnaies alternatives au Bitcoin (AltCoin).

Ces sociétés promettent de créer des monnaies avec un code protégé qui offre à la monnaie de nombreuses spécificités adaptées aux besoins du client. Vous souhaitez créer une devise qui sera utilisée uniquement sur un réseau de vente d'antiquités, par exemple? Ces sociétés peuvent créer la monnaie qui sera adaptée à votre plateforme. Les tarifs sont adaptés aux besoins du client.

Il existe également des générateurs automatiques de cryptodevises, mais ce sont généralement des arnaques. Le site [Bitcoin Exchange Guide](#) [11] explique que les générateurs comme Cryptocurrency Generator.pro, qui affirme générer des cryptomonnaies gratuitement, sont des « escroqueries pures et simples que tous les amateurs de cryptomonnaie devraient éviter ». Pour crédibiliser leur apparence, il est courant que ces générateurs aient recours à de faux témoignages et de faux numéros d'utilisateurs.

4.4 Réseau blockchain : "Mainnet & Testnets"

Chaque projet de blockchain doit être exécuté sur un réseau de blockchain pour assurer la fonctionnalité du projet ainsi que le transfert de sa propre monnaie numérique.

4.4.1 Testnets

Testnet, comme son nom l'indique, est un réseau alternatif destiné aux développeurs à des fins de test. Vous pouvez afficher un réseau de test en tant que réseau de démonstration à expérimenter. Il permet aux développeurs de mener des expériences sans gaspiller de l'argent réel. Les réseaux de test sont presque similaires aux réseaux principaux ou à la chaîne de blocs principale d'une crypto-monnaie. Le principe de fonctionnement des deux types de blockchains est similaire. Mais il y a aussi quelques différences clés. Un testnet est un réseau de démonstration alternatif. Ainsi, les cryptomonnaies sur un réseau de test n'ont aucune valeur sur le réseau et inversement.

Public Testnets (*Exemples : Ropsten, Kovan, Rinkeby, Görli*)

Local Testnets local, s'exécutant sur votre machine ou à petite échelle, sur des chaînes de blocs Ethereum privées.

Exemples : Ganache, eth-tester, clusters de réseaux de clients privés (par exemple, Geth avec un fichier de genèse personnalisé)

4.4.2 Mainnet

Mainnet est l'opposé complet du testnet. Mainnet est la blockchain principale d'un réseau de crypto-monnaie. Donc, quand quelqu'un dit Bitcoin mainnet, cela signifie la vraie blockchain Bitcoin.

Contrairement à testnet, qui est un réseau ouvert à des fins de test, mainnet est la vraie affaire. Pour Bitcoin et toutes les autres crypto-monnaies, les cryptomonnaies sur le réseau principal ont une valeur économique réelle. Les cryptomonnaies Mainnet sont celles dans lesquelles nous investissons et négocions régulièrement.

Comme les cryptomonnaies du réseau principal ont une valeur économique réelle, il sera très coûteux de tester les fonctionnalités et d'exécuter des tests sur le réseau. C'est pourquoi nous utilisons testnet pour exécuter nos tests avec de fausses crypto-monnaies n'ayant aucune valeur en dehors de testnet. Cela permet aux développeurs et aux passionnés de crypto-monnaie d'explorer la crypto-monnaie en détail. Lorsque les développeurs ont confiance en leur travail, ils le diffusent au grand public sous forme de réseau principal.

4.5 Analyse et Solutions possibles

The Difference Between Bitcoin, Ethereum, Ripple & Litecoin				
Cryptocurrency	Bitcoin	Ethereum	Ripple	Litecoin
Founder	Satoshi Nakamoto	Vitalik Buterin	Chris Larsen & Jed McCaleb	Charles Lee
Description	Decentralised cryptocurrency used in a peer-to-peer network	Ethereum - decentralised platform that runs smart contracts	Ripple is a real-time gross settlement system, currency exchange and remittance network by Ripple	Similar to Bitcoin - Litecoin is a peer-to-peer cryptocurrency and open source software project released under the MIT/X11 license
Founded	January 2009	January 2014	2012	October 2011
Market cap *	56%	13%	4%	3%
Maximum amount	21 million Bitcoins	18 million per year	100 billion Ripple units	84 million units
Transactions made via	Blockchain / mining	Decentralized programs or smart contracts	Ripple Labs issued currency	Mining based on uncomplex algorithm
Transaction time	10 minutes	12-14 seconds	4 seconds	2.5 minutes
Hashing Algorithm	SHA-256	Ethash	Ripple Protocol Consensus Algorithm (RPCA)	Scrypt
Mining	Processor intensive	Memory intensive	Not mined	Memory intensive
Price per USD **	BTC/USD: 17,571	ETH/USD: 610	XRP/USD: 0.32	LTC/USD: 332

* As at 13-December-2017: Source - <https://coinmarketcap.com/charts/>

** As at 12-December-2017

sepaforcorporates.com
@sepa4corporates

FIGURE 4.2 – Tableau comparatif des crypto-monnaies

Chaque cryptomonnaie de monnaie a sa propre blockchain, comme Bitcoin, Ethereum, Litecoin et la plupart des autres crypto-devises majeures dont nous avons entendu parler. Suivre cette voie implique la création de notre propre blockchain ou la création d'un fork (un doublon d'une blockchain existante),

Un jeton chevauche une chaîne de blocs existante, telle que Ethereum, et nécessite peu, voire aucune connaissance en matière de programmation, pour être créé, bien qu'on peut le rendre aussi complexe qu'on le souhaite.

Ethereum est probablement la plate-forme de jetons la plus populaire, ce qui signifie qu'il existe de nombreux outils pour vous aider à les créer et à les gérer sur ce réseau. C'est aussi l'une des blockchains les plus grandes et les plus fiables. Si vous souhaitez

effectuer du codage, vous devez apprendre Solidity (une variante de JavaScript), qui est plutôt un langage personnalisé pour Ethereum. Le choix d'Ethereum serait donc judicieux comme solution pour la création de notre cryptomonnaie iCubeCoin.

Nous pouvons également choisir NEO, qui est un peu moins convivial, mais prend en charge les langages de programmation courants tels que JavaScript et C ++. Une autre option est WAVES, qui affirme que plus de 13 000 jetons ont déjà été émis sur leur plate-forme et dont l'interface semble plutôt conviviale. Il y en a beaucoup d'autres qui existent et beaucoup d'autres en développement, mais pour l'instant, ce sont quelques-uns de nos meilleurs paris.

Quel réseau dois-je utiliser? S'il s'agit tout simplement de tester de petites transactions privées on peut essayer de commencer avec Ganache ou une chaîne privée Geth. Si l'on veut travailler avec un DApp, on peut essayer de faire le déploiement sur un réseau de test public adapté (Ex : Ropsten). Si l'on a une pleine foi et confiance en votre DApp on peut faire un déploiement sur le réseau principal(mainnet).

Chapitre 5

Solution proposée

5.1 Présentation de notre solution

Ethereum est devenu la première chaîne de blocs à offrir un service de création de jetons. Il offre un niveau de confiance exceptionnel en raison de sa maturité et de sa position forte sur le marché des crypto-devises. Tous les jetons construits sur Ethereum utilisent le standard ERC-20. La documentation est bien écrite et organisée, ce qui facilite le processus de développement.

Nous allons donc construire notre application décentralisée, ou dApp, sur la blockchain Ethereum. Le jeton de crypto-monnaie I3C se basera sur le standard de jeton ERC-20.

5.2 Réseau privé

Le projet iCubeCoin est un projet interne. Nous allons donc créer une blockchain privée ce qui peut présenter plusieurs avantages :

- S'assurer de la confidentialité des données et des transactions
- Pouvoir choisir les nœuds et les mineurs qui valident et surveillent les transactions de la blockchain
- Créer un token ou une cryptomonnaie dont le prix ne dépend pas de l'Ether (et donc limiter les coûts de transaction à la puissance des serveurs/ordinateurs connectés)

5.3 Jeton ERC-20

La blockchain Ethereum permet de créer notre propre crypto-monnaie, ou jeton, pouvant être acheté avec Ether, la crypto-monnaie native de la blockchain Ethereum.

ERC-20 est simplement un standard qui spécifie le comportement de ces jetons, de manière à ce qu'ils soient compatibles avec d'autres plates-formes telles que les échanges de crypto-monnaie.

Parmi ces standards, le plus connu et répandu est l'ERC-20, ce standard de token a été massivement utilisé et démocratisé par les levées de fonds de type ICO. L'avantage de ce standard est que la grande majorité des smart contracts et des DApps (applications décentralisées) peuvent interagir avec un token ERC-20 de manière native, sans avoir besoin de détails sur le token.

ERC signifie Ethereum Request for Comments. ERC20 définit 6 fonctions obligatoires que votre contrat intelligent doit implémenter, ainsi que 3 fonctions optionnelles.

Les fonctions optionnelles incluent :

- name
- symbols
- decimals

Les fonctions obligatoires sont :

- Total supply – la quantité de jetons qui existent actuellement
- Balance of – affiche le solde de l'adresse
- Transfer – envoie une certaine quantité de jetons à l'adresse
- Transfer from – utilisé pour échanger des jetons entre utilisateurs possédant ces jetons
- Approve – vérifie que l'adresse de votre portefeuille est éligible pour envoyer des jetons à un autre utilisateur
- Allowance – indique si un utilisateur dispose d'un solde suffisant pour envoyer des jetons à quelqu'un d'autre

5.4 Logique métier de notre projet

Les contrats intelligents sont l'endroit où réside toute la logique commerciale de notre application. C'est là que nous coderons la partie décentralisée de notre application. Les contrats intelligents sont chargés de la lecture et de l'écriture des données dans la blockchain, ainsi que de l'exécution de la logique métier.

5.4.1 Les acteurs

Le projet comporte deux principaux acteurs :

- **Les collaborateurs** : les acteurs de l'entreprise
- **Le service RH** : ceux qui gèrent le personnel

5.4.2 Les cas d'utilisation

1. Par un collaborateur
 - Créer son portefeuille pour faire parti du réseau
 - Créer un projet
 - Faire un Tip sur un projet existant
 - Solutionner un projet existant pour gagner des I3C
 - Transférer des I3C vers un autre portefeuille
 - Dépenser ses i3C à travers le kiosque
 - Récompenser le solutionneur d'un projet
2. Par le service RH
 - Valider l'inscription d'un collaborateur
 - Créer un projet
 - Valider un projet créer par un collaborateur
 - Mettre en place les produits et services du kiosque
 - Transférer des I3C vers un autre portefeuille
 - Récompenser le solutionneur d'un projet

5.4.3 Les fonctionnalités de notre application

1. Projet
 - Un projet crée doit être validé par le Service RH avant d'être actif
 - Deux type de projets seront traités :
 - projets à durée déterminé** nécessite une date de début et de fin.
 - projets récurrents** ne nécessite pas de date de début ni de fin.
2. Collaborateurs
 - À l'inscription, le collaborateur reçoit un certain nombre de jetons (définie préalablement par l'entreprise).
 - Un projet crée doit être validé par le Service RH avant d'être actif
 - Lorsqu'un collaborateur crée un projet (après validation du RH), il se voit octroyé un certains nombre de jetons à titre de récompense.
 - Un collaborateur peut décider d'augmenter la prime d'un projet en y ajoutant un certain nombre de jeton qui sera débiter de son solde (faire un TIP).
 - Lorsqu'un collaborateur participe à un projet, ce dernier se voit créditer sur son compte un certain nombre de jeton en guise de récompense; ce montant correspond à la prime du projet dont la répartition est gérée par le créateur du projet.

- Tous collaborateurs a le droit de créer un projet(Le projet ne peut être visible que s'il est actif).
- Un collaborateur peut transférer ses jetons à un autre sous contrainte de posséder ces jetons.
- ...

5.5 DApp

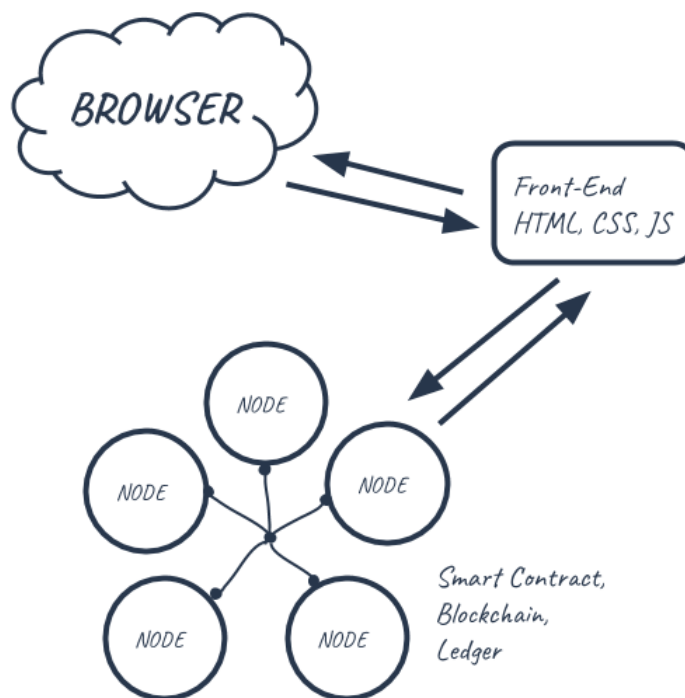


FIGURE 5.1 – Structure de notre dApp

Dans le cadre du stage, Nous mettrons en place un client frontal traditionnel écrit en **ReactJS**. Au lieu de parler à un serveur principal, ce client se connectera à une blockchain Ethereum locale que nous installerons. Nous coderons toute la logique commerciale relative à notre dApp dans un contrat intelligent Election avec le langage de programmation **Solidity**. Nous allons déployer ce contrat intelligent sur notre chaîne de chaînes Ethereum locale et permettre aux comptes de commencer à gagner et à dépenser leur I3C.

5.6 Web3.js

5.6.1 Interaction avec les SmartContracts

Pour appeler une fonction d'un Smart Contract , le serveur (simuler en local), va utiliser **Web3.js** pour effectuer des transactions sur la Blockchain. Une fois cette fonction appelée, Web3.js va ouvrir une "promise" et transmettra de manière asynchrone le résultat obtenu suite à cette requête.

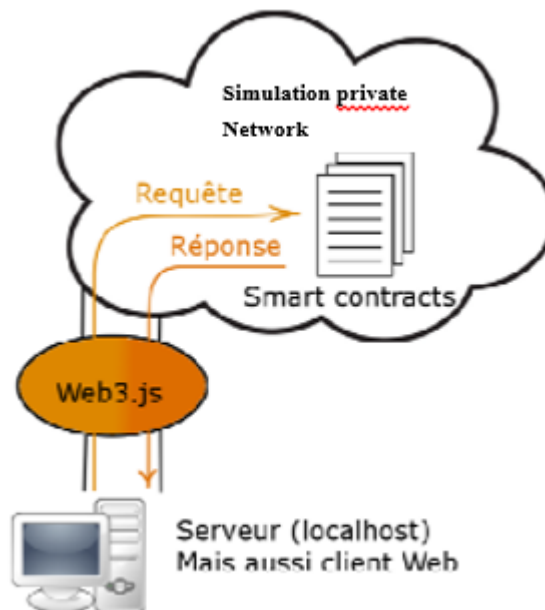


FIGURE 5.2 – Communication entre la plateforme et les Smart Contracts

5.6.2 Créer un portefeuille Ethereum avec Web3.js

Un wallet n'est qu'une application connectée à la blockchain, ayant une interface graphique utilisable facilement par les utilisateurs. C'est tout à fait possible d'en développer un en utilisant la librairie Web3js. Web3 possède de nombreuses fonctionnalités liées à la créations d'adresses, de sauvegarde, de transfert et de monitoring.

Pour stocker, recevoir, envoyer les tokens I3C , il faut a minima un compte. Le **web3.eth.accounts** contient des fonctions pour générer des comptes Ethereum et signer des transactions et des données.

5.7 Outils

Notre environnement de travail sera composé des outils suivants :

- **Geth (ou Go-ethereum)** Ethereum est écrit en utilisant le langage de programmation Go . Geth est l'outil / l'interface en ligne de commande permettant d'exécuter un nœud complet Ethereum.
- **NodeJS and NPM** NodeJS est une plate-forme JavaScript côté serveur permettant de créer des applications pouvant interagir avec votre nœud Ethereum. Lorsque nous installons NodeJS, nous installons également le Node Package Manager, ou NPM, requis pour configurer la plupart des outils et des bibliothèques nécessaires au développement de Ethereum.
- Framework **Truffle** qui nous permet de construire des applications décentralisées sur la blockchain Ethereum. Il fournit une suite d'outils nous permettant d'écrire des contrats intelligents avec le langage de programmation **Solidity**. Cela nous permet également de tester nos contrats intelligents et de les déployer dans la blockchain. Cela nous donne également un endroit pour développer notre application côté client.
- **Atom** un éditeur de texte pour écrire notre code.
- **Web3.js** - Un paquet JavaScript pour interagir avec le réseau Ethereum avec un navigateur Internet ou Node.js.

Côté client :

- **Navigateur web Chrome**
- **React Developer Toolset** une extension Chrome DevTools pour la bibliothèque JavaScript open source React. Il vous permet d'inspecter les hiérarchies des composants React dans les outils de développement de Chrome.

5.8 Informations de test

- Une levée de fond correspondant à **21000000 I3C** est alloué au projet
- À l'inscription, l'utilisateur reçoit automatiquement **1000000 I3C**,
- À chaque création de projet l'utilisateur porteur du projet reçoit automatiquement **1000 I3C**,

5.9 Environnement de travail

Nous travaillons sur un serveur **Windows Server 2016** par connexion distante.

Chapitre 6

Implémentations et Résultats

6.1 Implémentations

6.1.1 Installation de l'environnement de développement

6.1.1.1 Geth

Adresse de téléchargement : <https://geth.ethereum.org/downloads/> [12]

Version v1.8.7-stable de Windows

6.1.1.2 NodeJS and NPM

La commande "node -v" nous a permise de vérifier si node est déjà installé sur notre machine; si non on l'installe.

Adresse de téléchargement : <https://nodejs.org> [15]

Version : v10.15.3 (Version récente pour notre part)

6.1.1.3 Truffle

Nous avons installer Truffle en tant qu'un paquetage de Node. Nous avons en premier lieu désinstaller toute ancienne version potentielle qu'il pouvait avoir. Nous avons entré les commandes suivantes dans le terminal :

Commande de désinstallation : `npm uninstall -g truffle`

Commande d'installation : `npm install -g truffle@4.0.4`

Nous avons ajouter une version ici "@ 4.0.4". (En effet, nous ne souhaitons pas installer la dernière version susceptible d'apporter des modifications radicales du fait que nous suivons un tutoriel avec des codes sources à tester).

Commande de vérification : `truffle version` (la commande envoie comme résultat la version 4.0.4 pour Truffle et 0.4.18 pour Solidity)

6.1.1.4 Atom

Adresse de téléchargement : <https://github.com/atom/atom>[16]

Version : v1.35.0 (Version récente pour notre part)

Installation extension language-ethereum : `apm install language-ethereum`

Atom Package Manager (APM) est le gestionnaire de paquets Atom. la commande "`apm -v`" permet de voir sa version. Atom Package Manager est donc très utile pour installer des extensions et des plugins pour l'éditeur de texte Atom. Et dans ce cas, nous l'utilisons pour installer l'extension **language-ethereum**, qui ajoute la prise en charge du langage Solidity, que nous utiliserons pour rédiger nos contrats intelligents.

6.1.1.5 Git

Adresse de téléchargement : <https://git-scm.com/download/win>[17]

Version : v2.21.0

6.1.1.6 React Developer Tools

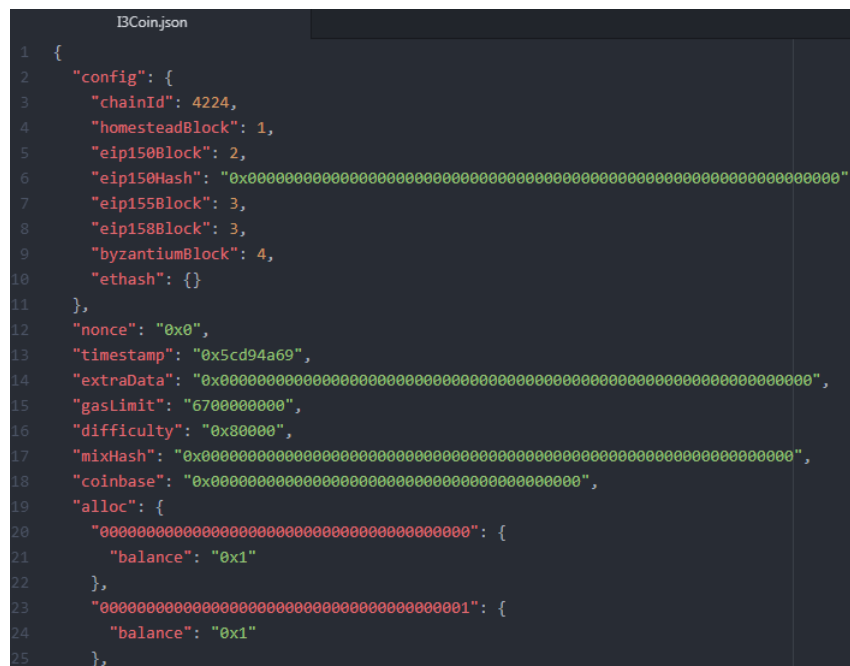
Ajouter l'extension <https://chrome.google.com/webstore/detail/react-developer-tools/fmkadmapgofadopljbjfkapdkoienihi?hl=en> [18] au navigateur chrome

6.1.2 Créer une blockchain privée

6.1.2.1 Création d'un bloc de genèse

La création d'un bloc de genèse à partir de zéro peut s'avérer pénible et la réutilisation d'un bloc existant peut poser problème si sa version est plus ancienne que notre installation actuelle de Geth. Pour s'assurer que notre fichier de bloc de genèse est compatible avec notre version de Geth, nous le créerons à l'aide d'un outil appelé **Puppeth**. Puppeth est un outil de ligne de commande installé avec Geth. Il fournit des fonctionnalités permettant de créer et de surveiller notre propre réseau Ethereum.

1. Se positionner dans un repertoire **private** que nous avons créer et exécutez la commande : **Puppeth**
2. Choisir le nom du réseau. Dans notre cas "i3Coin"
3. Parmi les options proposées, utiliser la commande #2 pour configurer une nouvelle genèse.
4. Répondre à une série de question :
 - *"Quel moteur de consensus à utiliser?"*. Nous avons le choix entre une preuve de travail et une preuve d'autorité. Nous allons donc utiliser la première option, Ethash qui est la preuve de travail..
 - *"Quels comptes doivent être pré-financés?"* Nous devons choisir une chaîne ou un identifiant de réseau pour notre réseau. L'important est que cet ID soit différent des ID déjà utilisés pour les réseaux publics. L'ID n° 1 est donc réservé au réseau principal. L'ID n° 2 est réservé au réseau de test Morden, devenu obsolète. Le numéro 3 concerne le réseau de test Ropsten. Le numéro 4 est pour Rinkeby, un autre réseau de test public que nous utiliserons plus tard. Et 42 est pour Kovan, un autre réseau de test. Nous utiliserons l'ID de réseau 4224.
 - *"Quelque chose d'autre à intégrer dans le bloc de la genèse?"* On répond : "Non".
5. Nous avons donc créé la genèse en mémoire. Pour l'exporter dans un fichier JSON, nous choisissons l'option #2, "Gérer la genèse existante", puis l'option n° 2 à nouveau pour exporter la configuration de la genèse. Nous pouvons choisir un nom pour le fichier ou conserver simplement la valeur par défaut : "I3Coin.json".



```

I3Coin.json
1  {
2    "config": {
3      "chainId": 4224,
4      "homesteadBlock": 1,
5      "eip150Block": 2,
6      "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000",
7      "eip155Block": 3,
8      "eip158Block": 3,
9      "byzantiumBlock": 4,
10     "ethash": {}
11   },
12   "nonce": "0x0",
13   "timestamp": "0x5cd94a69",
14   "extraData": "0x0000000000000000000000000000000000000000000000000000000000000000",
15   "gasLimit": "670000000",
16   "difficulty": "0x80000",
17   "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
18   "coinbase": "0x0000000000000000000000000000000000000000000000000000000000000000",
19   "alloc": {
20     "0000000000000000000000000000000000000000000000000000000000000000": {
21       "balance": "0x1"
22     },
23     "0000000000000000000000000000000000000000000000000000000000000001": {
24       "balance": "0x1"
25     },

```

FIGURE 6.1 – Contenu du fichier I3Coin.js

Nous allons expliquer quelques éléments intéressants de ce fichier de genèse.

Le fichier commence par le bloc **«config»** qui contient tous les paramètres de configuration et les seuils contrôlant les opérations de base du réseau. Lire la suite ici .

chainId : protège le réseau contre les attaques par rejeu . Il agit comme un décalage pour empêcher les attaquants de déchiffrer des valeurs continues sur votre réseau. **homesteadBlock :** Homestead est la deuxième version majeure d'Ethereum (la première version est Frontier). La valeur 0 signifie que vous utilisez cette version.

difficulté Cela détermine à quel point il est difficile d'exploiter votre réseau. Lorsque vous développez sur un réseau de test, définissez-le sur la valeur la plus basse possible pour ne pas avoir à attendre trop longtemps pour la validation de blocs.

gasLimit Ceci définit la limite de coût en gaz par bloc. Définissez cette valeur sur high pour éviter les limitations lors des tests.

nonce & mixhash Nonce et mixhash sont des valeurs qui, lorsqu'elles sont combinées, permettent de vérifier qu'un bloc a été réellement exploité de manière cryptographique et qu'il est donc valide. Le mixhash est un hachage de 256 bits, ce qui prouve, lorsqu'il est combiné avec le nonce de 64 bits, qu'une quantité suffisante de calculs a été effectuée sur ce bloc : la preuve de travail.

alloc Permet de définir une liste de portefeuilles pré remplis. C'est une fonctionnalité spécifique à Ethereum pour gérer la période de «pré-vente Ether». Vous pouvez également laisser cela comme un hachage vide.

6.1.2.2 Initialiser un noeud privé

Commande : **geth - -datadir . init I3Coin.json**

Le nœud privé sera créé à l'aide de la commande geth avec les paramètres suivants. "datadir" sera utilisé pour définir le répertoire cible dans lequel les données du nœud privé seront sauvegardées et nous utilisons également la commande **"init"** pour spécifier le bloc de genèse permettant d'initialiser le nœud privé.

6.1.2.3 Créer un compte

Commande : **geth - -datadir . account new**

Le nouveau compte est verrouillé par un mot de passe. Chaque compte aura donc une clé privée. Cette clé privée sera protégée par un mot de passe. Elle ne pourra donc être utilisée que pour signer une transaction avec le mot de passe. Par défaut, le premier compte, donc celui avec l'indice 0 sera celui qui recevra toutes les récompenses du minage lorsque nous commencerons à faire le minage

6.1.2.4 Lancer notre blockchain

Maintenant que notre blockchain est créée, il faut pouvoir la partager et permettre à des utilisateurs d'interagir avec cette dernière et pouvoir miner (écrire) des nouveaux blocs. Nous allons créer un script *startnode.cmd* qui définira tous les paramètres requis pour nous permettre d'interagir avec notre nœud privé. Le script contient la commande suivante :

```
geth --networkid 4224 --mine --minerthreads 1 --datadir "." --nodiscover --rpc --rpcport "8545"
--port "30303" --rpccorsdomain "*" --nat "any" --rpcapi eth,web3,personal,net --unlock 0 --password ./password.sec
```

FIGURE 6.2 – fichier script pour le lancement de la blockchain

Commande d'exécution du script : **./startnode.cmd**

Quelques informations :

- Le network id correspond au "réseau de la Blockchain". Plusieurs Blockchain peuvent coexister au sein de ce réseau
- rpc : signifie qu'on ouvre une api permettant d'interagir avec notre Blockchain
- rpcapi : lancement de plusieurs API

6.1.3 Lancement de notre propre jeton ERC20 : I3Coin (utilisation de OpenZeppelin)

Nous allons créer notre jeton ERC-20 Ethereum I3C avec un contrat intelligent. Les contacts intelligents sont écrits dans un langage de programmation appelé **Solidity**.

On crée et on initialise le dossier de notre projet Truffle :

```
1 $ mkdir I3Coin
2 $ cd I3Coin
3 I3Coin $ truffle init
4 I3Coin $ npm init
```

Listing 6.1 – Création du projet ICubeCoin

Pour créer notre Token, nous allons utiliser la Bibliothèque **OpenZeppelin**.

OpenZeppelin est une bibliothèque pour le développement sécurisé de contrats intelligents. Il fournit des implémentations de normes telles que ERC20 et ERC721 que

vous pouvez déployer tel quel ou à adapter à vos besoins, ainsi que des composants Solidity pour la création de contrats personnalisés et de systèmes décentralisés plus complexes.

Nous allons donc importer OpenZeppelin comme suit :

```
$ npm install openzeppelin-solidity@1.12.0
```

Nous allons à présent créer notre contract token en créant un nouveau fichier I3Coin.sol contenant le code suivant :

```
pragma solidity ^0.4.18;

import 'zeppelin-solidity/contracts/token/ERC20/StandardToken.sol';

contract I3Coin is StandardToken {

    //Construction du jeton I3Coin (I3C)
    string public constant name = 'ICubeCoin';
    string public constant symbol = 'I3C';
    uint8 public constant decimals = 2;
    uint constant _initial_supply = 2100000000;

    //Constructeur
    function I3Coin() public {
        //newUtilisateur.nom="Nathalie";
        totalSupply_ = _initial_supply;
        balances[msg.sender] = _initial_supply;
        Transfer(address(0), msg.sender, _initial_supply);
    }
}
```

FIGURE 6.3 – Création du token ICubeCoin

6.1.4 Implantation de la logique métier

Nous allons définir le comportement attendu de notre contrat. Le but est de faire gagner des cryptomonnaies et de les dépenser.

6.1.4.1 Variables et types de données

Nous avons :

- des mappings
- des structures
- des énumération

énumération Nous déclarons une énumération **TypeOperation**, comme cela existe dans plusieurs langages de programmation. Celle-ci contient les différents types d'opé-

ration : CREATION_PROJET, VALIDATION_PROJET, TIP, SOLUTIONNER_PROJET, TRANSFERT, CREATION_COMPTE

```
enum TypeOperation{CREATION_PROJET,VALIDATION_PROJET,TIP,SOLUTIONNER_PROJET,TRANSFERT,CREATION_COMPTE}
```

FIGURE 6.4 – Déclaration d’une variable de type énumération

structure Nous déclarons 3 structures :

- **Utilisateur** : celle qui contiendra les informations non sensibles de l'utilisateur
- **Projet** : celle qui contiendra les informations du projet
- **Kiosque** : celle qui contiendra les information d'un produit/service

```
struct Utilisateur {  
    uint id;  
    string mail;  
    string nom;  
    uint actif;  
    uint profil;  
}  
  
struct Projet {  
    uint id;  
    address createur;  
    string name;  
    string description;  
    uint256 price;  
    uint etat;  
    uint256 priceSolution;  
}  
  
struct Kiosque {  
    uint id;  
    string name;  
    string description;  
    uint256 price;  
    address createur;  
    uint etat;  
}
```

FIGURE 6.5 – Déclaration des structures du projet

mapping Les *utilisateurs*, *projets* et *kiosques* sont stockés dans les variables *listeUtilisateur*, *projets*, *kiosques* de type mapping.

```
mapping (address => Utilisateur) public listeUtilisateur;  
mapping (uint => Projet) public projets;  
mapping (uint => Kiosque) public kiosques;
```

FIGURE 6.6 – Déclaration des variables de type mapping

événements La blockchain stocke les transactions en blocs. Chaque transaction a des journaux, représentant les événements Solidity qui se sont produits.

Nous avons plusieurs événements qui se déclencheront lorsque certaines actions (fonctions) sont posées.

```
event LogCreerProjet(  
    uint indexed _id,  
    address indexed _createur,  
    string _name,  
    uint256 _price  
);  
  
event LogValiderProjet(  
    uint indexed _id,  
    address indexed _createur,  
    string _name,  
    uint256 _price,  
    uint256 _recompense,  
    address indexed _valideur  
);  
  
event LogSolutionnerProjet(  
    uint indexed _id,  
    address indexed _solutionneur,  
    uint _idProjet,  
    string _nomProjet,  
    uint256 _montant,  
    address indexed _createur  
);  
  
event LogTip(  
    uint indexed _id,  
    address indexed _tippeur,  
    uint _idProjet,  
    string _nomProjet,  
    uint256 _montant,  
    address indexed _createur  
);  
  
event LogTransfert(  
    uint indexed _id,  
    address indexed _envoyeu,  
    address indexed _receveur,  
    uint256 _montant  
);  
  
event LogKiosque(  
    uint indexed _idProduit,  
    address indexed client,  
    string _nomProduit,  
    uint256 _prixProduit,  
    address _vendeur  
);
```

FIGURE 6.7 – Déclaration des évènements du projet

6.1.4.2 Les fonctions de notre contrat

Dans Solidity un contrat peut en réalité être considéré comme une classe, comme on en trouve en programmation orientée objet. C'est-à-dire qu'à partir d'une classe on peut créer plusieurs objets (les instances de la classe), et que chacun aura son propre espace mémoire pour stocker les valeurs des variables que nous avons déclarées dans le paragraphe précédent.

Ainsi de la même manière que dans les autres langages orientés objets, on peut définir pour un contrat des méthodes, qui sont plutôt nommées fonctions dans Solidity.

Et l'une d'elles est particulière puisque c'est elle qui est automatiquement appelée à la création (ou plutôt l'instanciation) d'un contrat : le constructeur. Comme dans beaucoup de langages, elle a pour signe distinctif d'avoir un nom identique au contrat :

```
//Constructeur
function I3Coin() public {
    totalSupply_ = _initial_supply;
    balances[msg.sender] = _initial_supply;
    Transfer(address(0), msg.sender, _initial_supply);
    Utilisateur memory u;
    utilisateurCounter++;
    u.mail="rh@bp.fr";
    u.nom="rh";
    u.id=utilisateurCounter;
    u.actif=1;
    u.profil=1;
    listeUtilisateur[msg.sender] = u;
}
```

FIGURE 6.8 – Constructeur du projet

Passons aux fonctions publiques. Nous allons citer quelques unes :

```
function ajouterUtilisateur (string _mail, string _nom,
address _addr) public {
    utilisateurCounter++;
    Utilisateur memory nouvelUtilisateur;
    nouvelUtilisateur.mail=_mail;
    nouvelUtilisateur.nom=_nom;
    nouvelUtilisateur.actif=0;
    nouvelUtilisateur.profil=0;
    nouvelUtilisateur.id=utilisateurCounter;
    listeUtilisateur[_addr] = nouvelUtilisateur;
}

function creerProjet(string _name, string _description, uint256 _price) public {
    projetCounter++;
    operationCounter++;
    Projet memory p;
    p.id = projetCounter;
    p.name=_name;
    p.description=_description;
    p.price = _price;
    p.createur = msg.sender;
    p.etat = 0;
    p.priceSolution = 0;
    projets[projetCounter] = p;
    LogCreerProjet(projetCounter, msg.sender, _name, _price);
}

function tipProjet(uint _id,uint256 _montant) public {
    require(projetCounter > 0);
    require(_id > 0 && _id <= projetCounter);
    Projet storage projet = projets[_id];
    balances[msg.sender]-=_montant;
    balances[projet.createur]+=_montant;
    projet.price+=_montant;
    LogTip(operationCounter,msg.sender,_id,projet.name,_montant,projet.createur);
}
```

FIGURE 6.9 – Fonctions du projet

6.1.5 Lancement de notre smart contract

Configuration de réseau :



```
truffle-config.js
const path = require("path");

module.exports = {
  // See <http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!
  networks: {
    ganache: {
      host: "localhost",
      port: 7545,
      network_id: "*"
    },
    i3coin: {
      host: "localhost",
      port: 8545,
      network_id: "4224",
      gas: 6700000
    }
  },
  contracts_build_directory: path.join(__dirname, "app/src/contracts"),
};
```

FIGURE 6.10 – Configuration du réseau de l'exécution du smart-contract

Commande : **truffle migrate --compile-all --reset --network i3coin**

6.1.6 Conception - Coté Client

6.1.6.1 Template ReactJS

Nous avons choisi le template **Shards Dashboard React** téléchargeable sur GitHub à l'adresse <https://github.com/DesignRevision/shards-dashboard-react> [19].

C'est un pack de modèles de tableaux de bord d'administration React gratuit comprenant un système de conception moderne et de nombreux modèles et composants personnalisés. Il n'est donc pas nécessaire d'être un expert front-end pour réaliser la vue de notre dApp.

Nous avons créer un dossier *front-end* dans le dossier de notre projet i3Coin et y ajouter notre template. Nous allons par contre réadapter le template en fonction de notre projet i3Coin en modifiant les composants des vus etc...

6.1.6.2 Configurer web3.js

Web3.js est une bibliothèque javascript qui permet à notre application côté client de communiquer avec la chaîne de blocs. Nous allons tout d'abord intégrer web3.js dans le projet avec : **npm install web3**

Après avoir importé Web3 dans notre fichier App.js , nous avons créer une instance web3, en définissant le fournisseur HTTP :

```
this.web3Provider = new Web3.providers.HttpProvider('http://localhost:8545');
App.web3 = new Web3(this.web3Provider);
this.i3Coin = TruffleContract(I3Coin)
this.i3Coin.setProvider(this.web3Provider)
```

FIGURE 6.11 – Configuration de web3.js

6.1.7 Lancement de l'application

Le lancement de l'application se fait avec la commande suivante : **npm start**

6.1.8 Difficultés rencontrées

La réalisation d'un tel projet, résolument novateur, n'a pas été sans embûche, ni sans découvertes et enseignements de toutes sortes.

6.1.8.1 L'authentification par adresse mail et mot de passe

Problème Comment effectuer l'authentification ? Pour se connecter à la blockchain , il faut déverrouiller le compte avec la commande : **web3.personal.unlockAccount("<adresse du compte>", "<pass>")** Or l'utilisateur ne renseigne que : l'adresse email + pass

Solution Nous avons mis en dure dans le code les informations (adresse + pass) d'un compte utilisateur qui ira chercher la correspondance de l'adresse avec l'email.

6.1.8.2 Payer les transactions avec les éthers

Problème De part la configuration du bloc de genèse avec le choix du consensus : **Preuve de travail**, chaque compte doit payer ses transactions en éther. Chaque compte doit donc posséder un certain nombre d'éther pour effectuer une transaction. Comment distribuer les éthers aux différents comptes ?

Solution créer notre propre robinet qui envoie gratuitement des milliards de cette éther de chaîne privée. En effet dans notre cas de test, l'éther n'a aucune valeur réelle. Une adresse (**Coinbase**) sera défini pour recevoir toutes les récompenses du minage. Elle se chargera par la suite de redistribuer les éthers aux autres adresses en cas de besoin.

6.2.1 Lancement de la blockchain

[illegible]

FIGURE 6.12 – Lancement de la blockchain

6.2.2 Présentation de notre application

Nous allons décrire les écrans développés de notre application ainsi que les différentes fonctionnalités.

6.2.2.1 Authentication

L'utilisateur saisit son adresse email et son mot de passe et appuie sur 'connexion' :

- si l'authentification est valide, alors l'utilisateur est redirigé vers l'écran d'Accueil
- sinon un message d'erreur s'affiche Si l'utilisateur ne dispose pas de compte, alors il peut s'inscrire en cliquant sur le bouton 'Créer un compte' pour accéder au formulaire d'inscription.

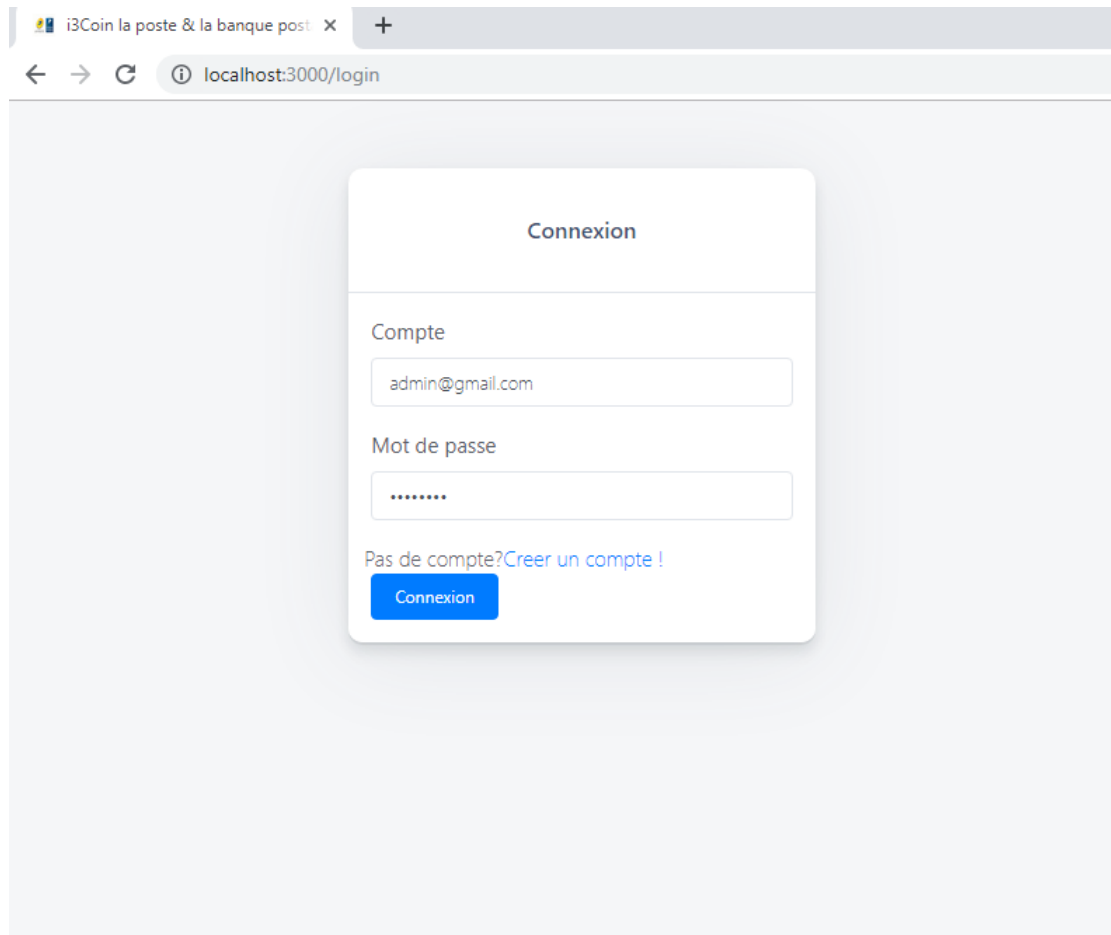
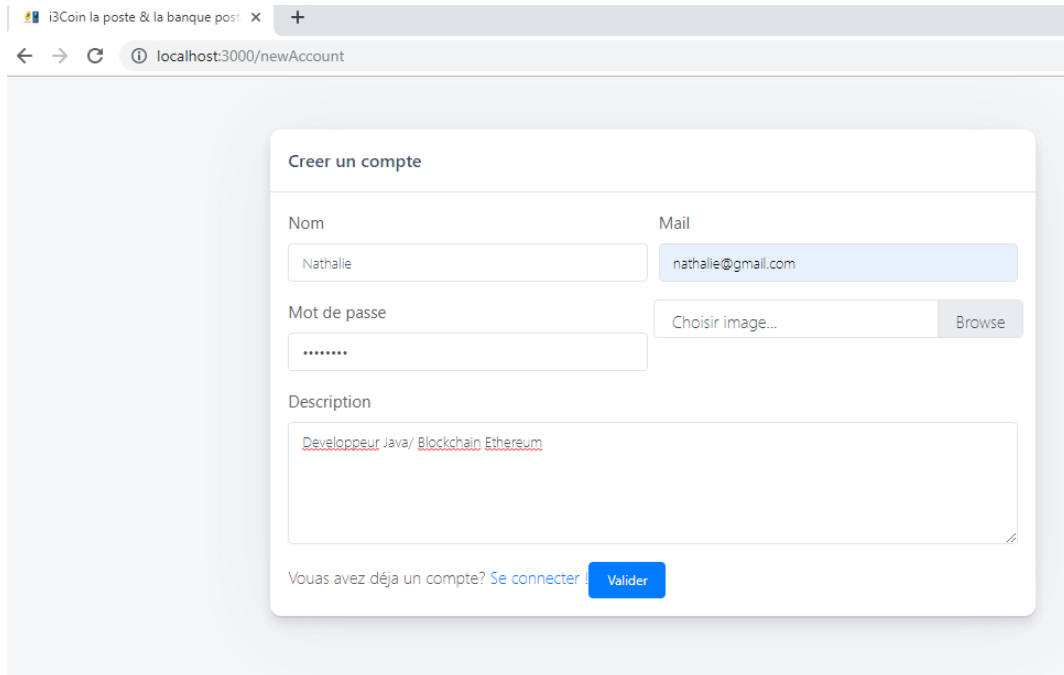


FIGURE 6.13 – page d’authentification

6.2.2.2 Création d’un compte

Pour s’inscrire, l’utilisateur doit saisir son nom , son adresse email et son mot de passe . En cliquant sur ‘Valider’, l’utilisateur crée un compte. En effet l’application crée en arrière plan un compte de monnaie iCubeCoin (adresse du compte + clé privé). Mais ce compte doit être validé par une autorité compétente (le service RH) avant que l’utilisateur ne puisse se connecter.



Créer un compte

Nom: Nathalie

Mail: nathalie@gmail.com

Mot de passe: *****

Choisir image... Browse

Description: Developpeur Java/ Blockchain Ethereum

Vous avez déjà un compte? [Se connecter](#) Valider

FIGURE 6.14 – page de création d’un compte

6.2.2.3 Menu

le coté gauche des pages contient un Menu. Le menu comprend :

- **Accueil**
- **Profil utilisateur**
- **Gestion des utilisateurs** *Uniquement visible pour le service RH*
- **Gestion kiosque** *Uniquement visible pour le service RH*
- **Mes projets**
- **Gestion des projets**
- **Transfert**
- **Kiosque** *Uniquement visible pour les collaborateurs de l’entreprise*

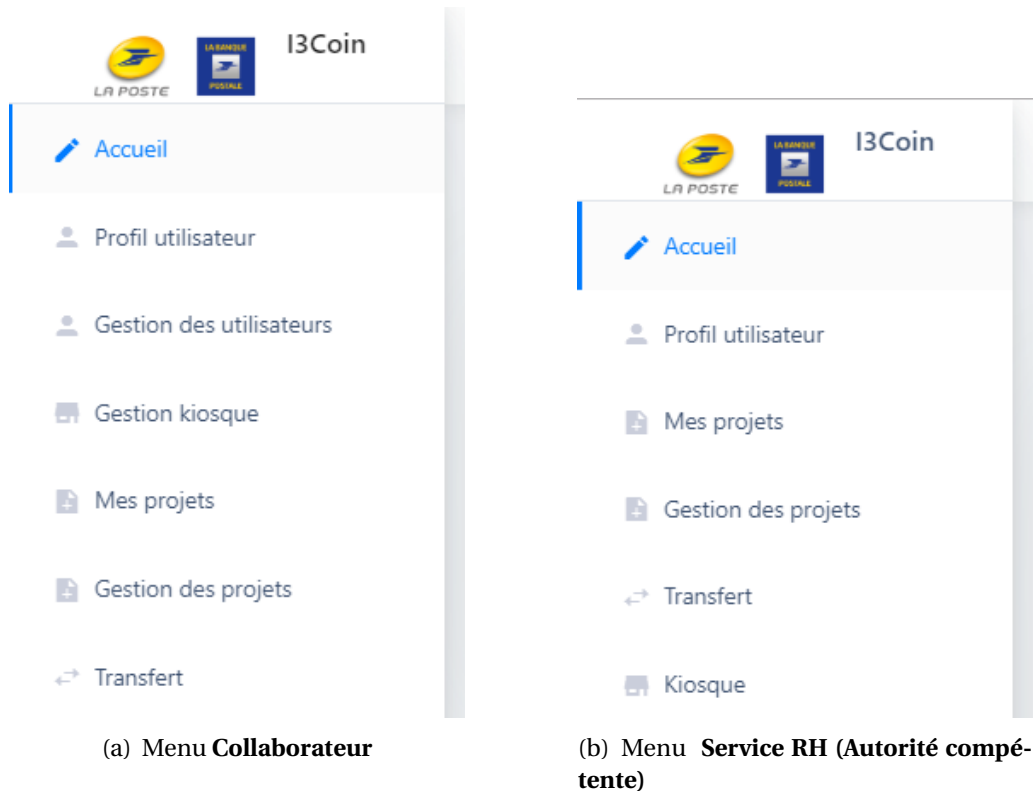


FIGURE 6.15 – Menu à gauche des pages

6.2.2.4 Accueil

L'écran d'accueil de l'application présente une liste de toutes les transactions entrantes et sortantes du compte connecté. Pour chaque transaction, la liste de transactions affiche les données suivantes : le type de transaction , le montant et les intervenants de cette transaction.

Type de transactions :

- création d'un compte
- création d'un projet
- tip sur un projet
- solutionner un projet
- transfert de i3C
- Achat/Vente des services ou produits du kiosque

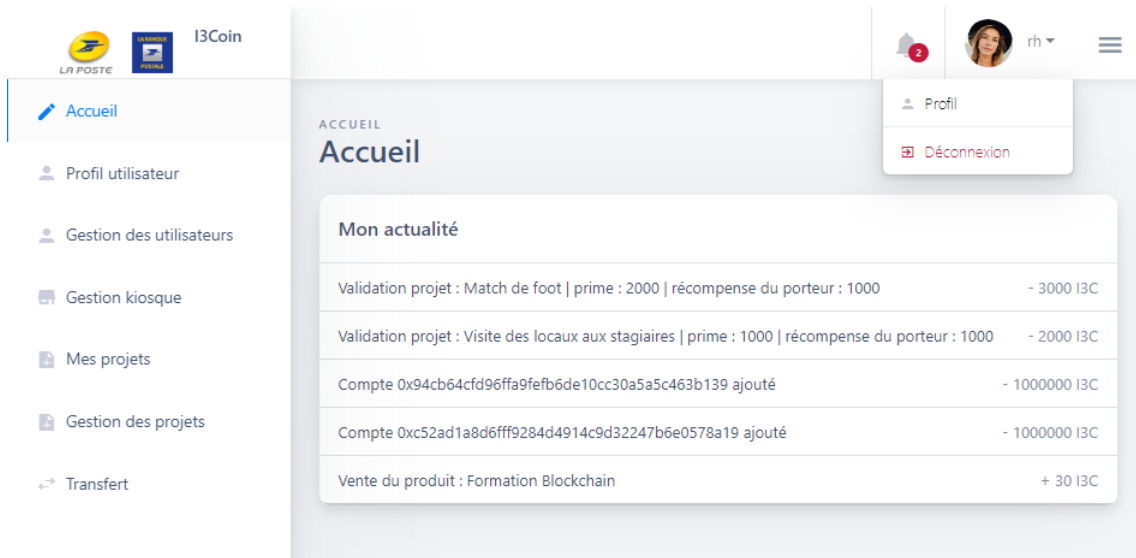


FIGURE 6.16 – page d'accueil

6.2.2.5 Profil utilisateur

La page **Profil utilisateur** présente les informations de l'utilisateur connecté :

- Nom
- Adresse mail
- la clé publique, soit l'adresse du compte.
- le solde du compte en I3C.
- Photo (Nous n'avons pas géré l'enregistrement des fichiers images dans le projet)

Les informations telles que : **Nom**, **adresse mail**, **photo** sont modifiables dans le formulaire à droite.

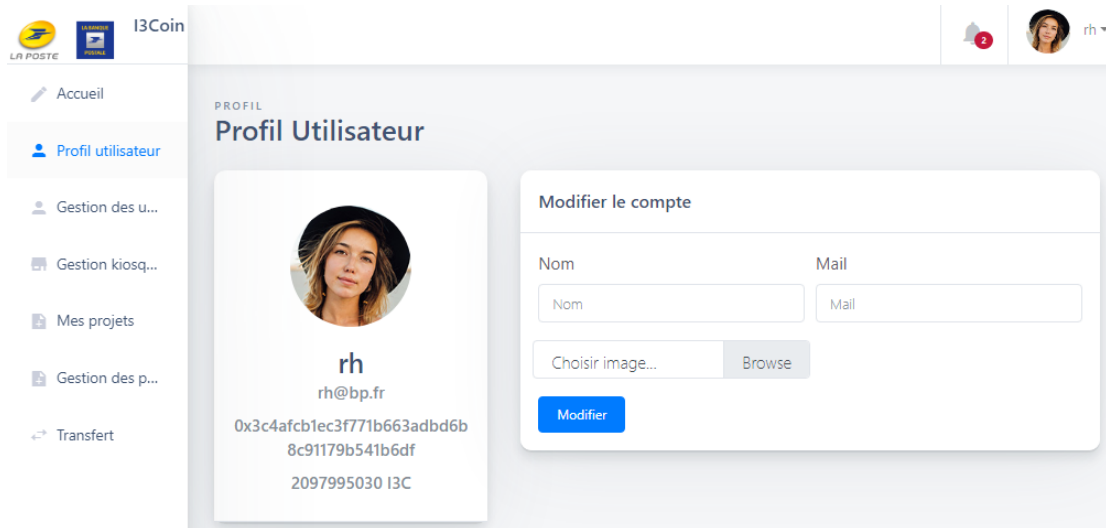


FIGURE 6.17 – page - profil utilisateur

6.2.2.6 Mes Projets

Cet écran affiche la liste des projets créés par l'utilisateur connecté. Il peut :

- Faire un tip sur le projet en cliquant sur le bouton **tip**
- Récompenser un solutionneur (une personne ayant participé au projet) en cliquant sur le bouton **solutionner**

La création d'un projet nécessite la validation du projet auprès du service RH.

- Les projets créés et qui ne sont pas encore validés sont marqués : **En cours de validation**, Il n'y a donc pas de possibilité de faire un **Tip** ou de **Solutionner** le projet; d'où les boutons grisés.
- Les projets déjà validés sont marqués : **Validé** . Il y a donc possibilité de faire un **Tip** ou de **Solutionner** le projet

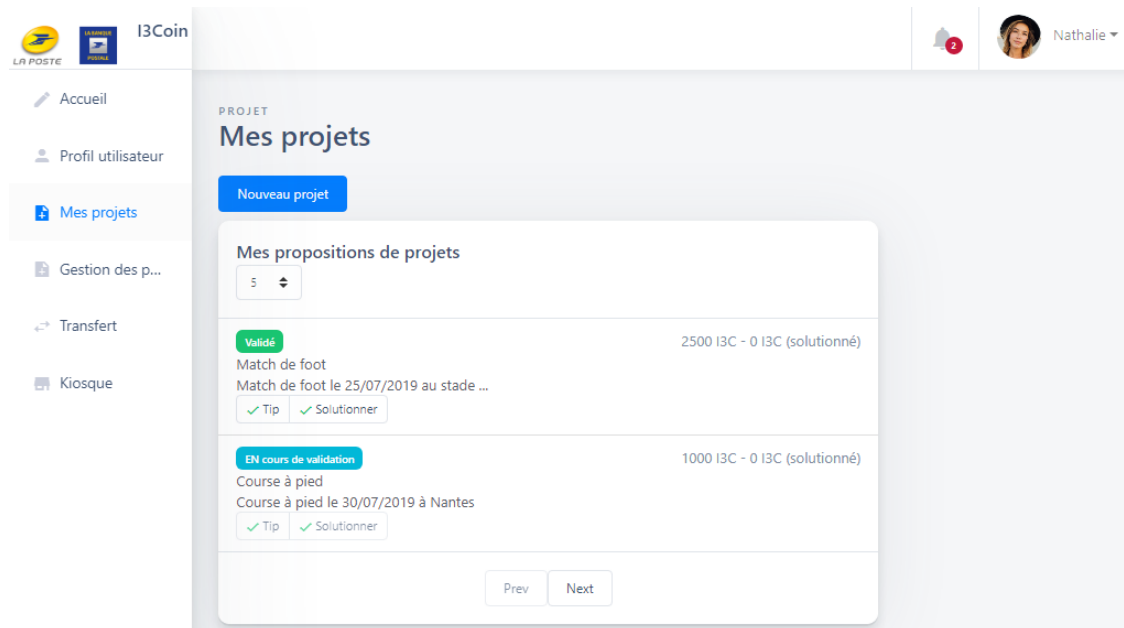


FIGURE 6.18 – page - liste des projets

Nouveau Projet

Pour ajouter un projet, l'utilisateur Porteur doit cliquer sur le bouton "**nouveau projet**" et renseigner les champs du formulaire suivants :

- Titre du projet : le nom/libellé du projet
- Prime : la prime associée au projet
- Description : les détails sur le projet

L'utilisateur doit cliquer sur 'Valider' pour valider sa saisie.

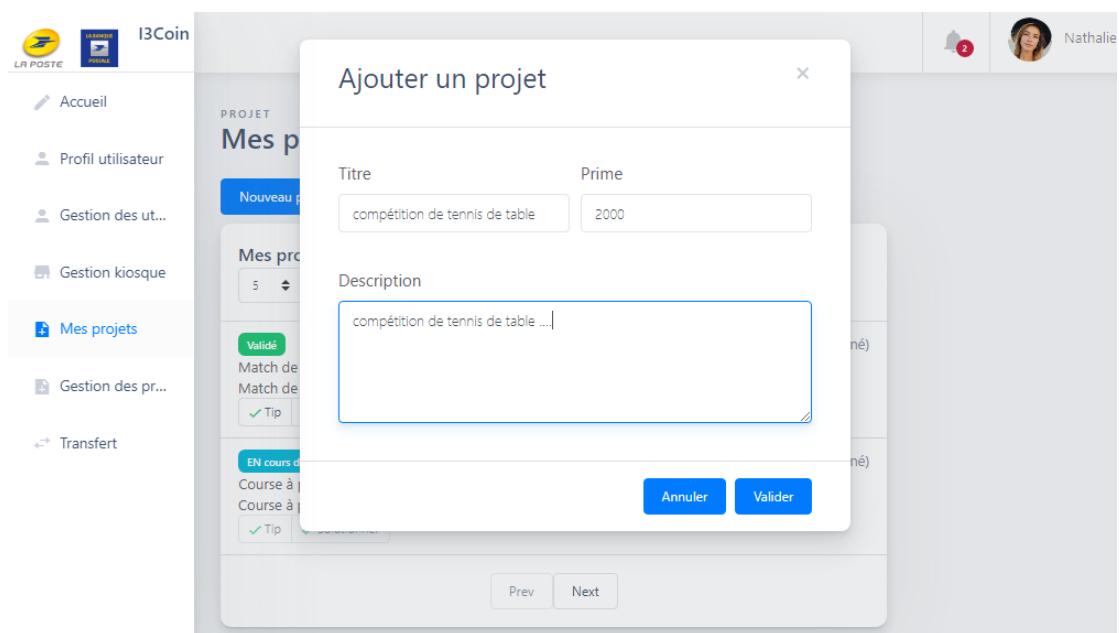


FIGURE 6.19 – page - création d'un nouveau projet

Tip

En cliquant sur le bouton **tip** d'un élément de la liste des projets validés, une popup s'affiche, donnant accès à un formulaire. L'utilisateur devra entrer le montant du tip puis cliquer sur **Valider** pour valider la transaction ou cliquer sur **Annuler** pour l'annuler.

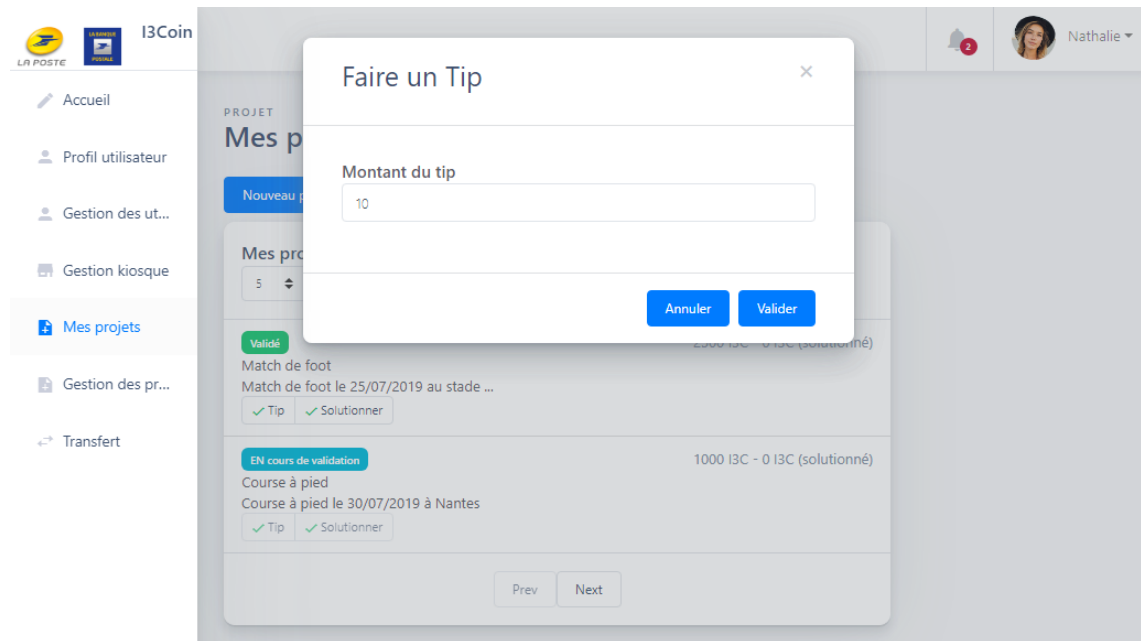


FIGURE 6.20 – page - Faire un tip sur un projet

Solutionner un projet

En cliquant sur le bouton **solutionner** d'un élément de la liste des projets validés, une popup s'affiche, donnant accès à un formulaire. L'utilisateur devra entrer l'adresse du solutionneur, le montant de sa récompense et une note (en option), puis cliquer sur **Valider** pour valider la transaction ou cliquer sur **Annuler** pour l'annuler.

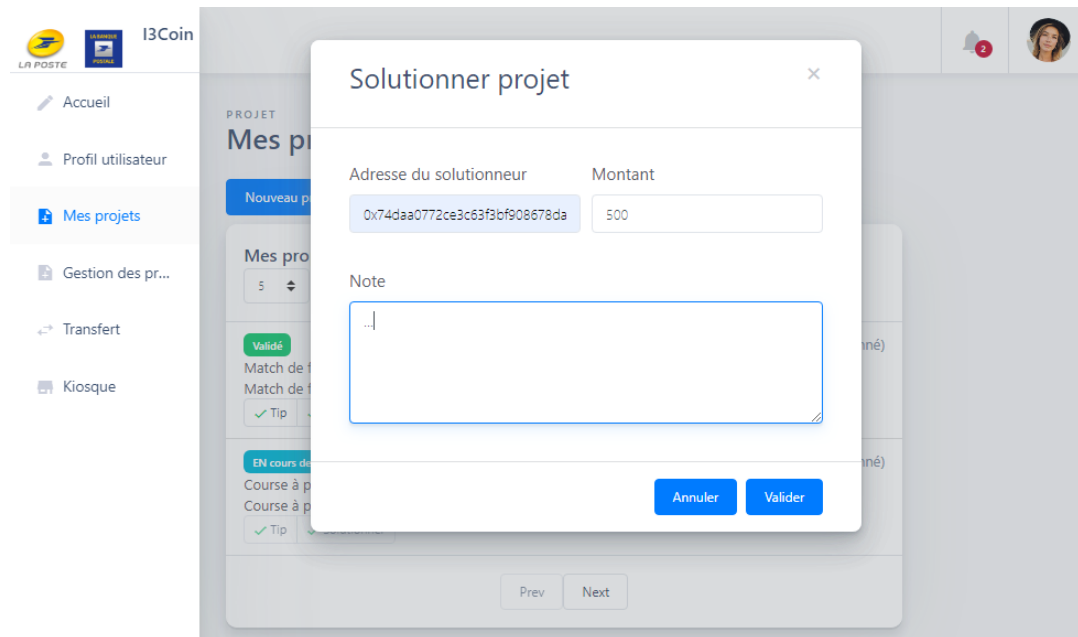


FIGURE 6.21 – page - Solutionner un projet

6.2.2.7 Gestion des Projets

Cet écran affiche la liste de tous les projets créés. Tous les projets ajoutés par chacun des collaborateurs ou du service RH sont visibles par tous. De même que précédemment, l'état des projets est renseigné (validé / en cours de validation).

Les actions présentes sur cette page sont les suivantes :

- Valider un projet en cliquant sur le bouton **Valider**. Cette action n'est visible que par le service RH qui est l'autorité compétente ayant ce droit de validation.
- Faire un tip sur le projet en cliquant sur le bouton **Tip** (Le processus étant le même que pour la page **Mes projet**)
- Mettre un projet en favoris en cliquant sur le bouton **Favoris**

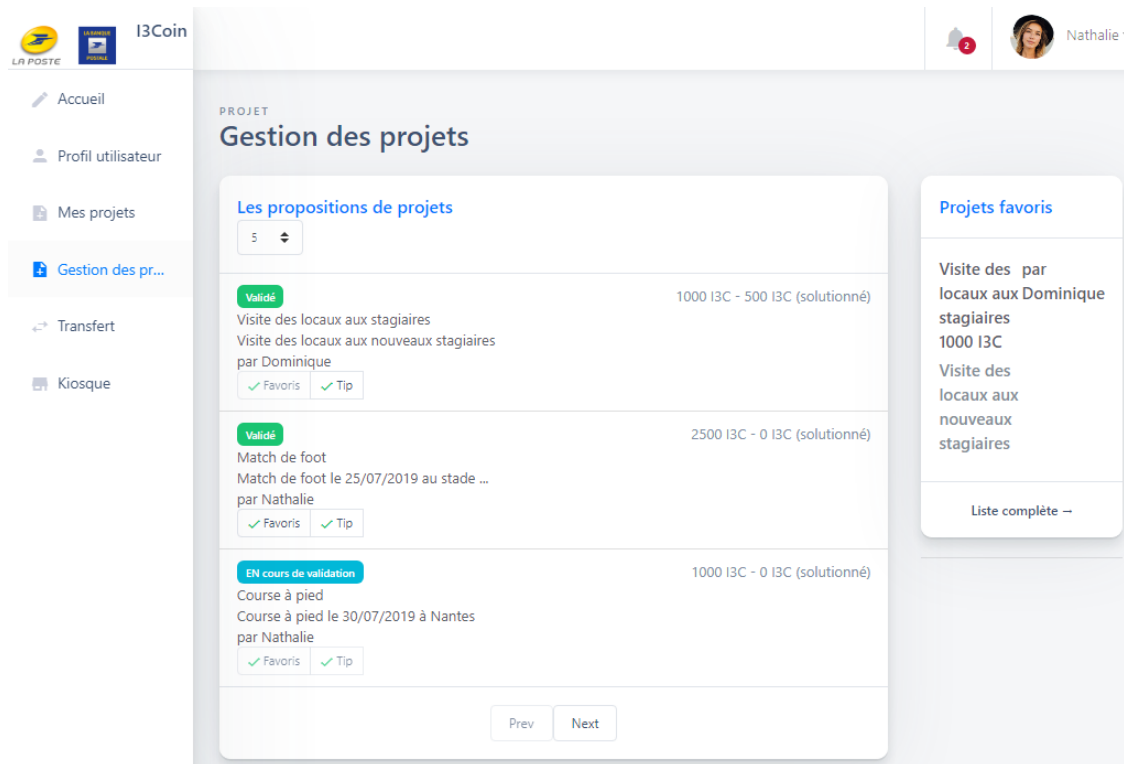


FIGURE 6.22 – page - gestion des projets

Valider un projet

Pour Valider un projet, l'utilisateur clique sur le bouton "**Valider**" d'un élément de la liste des projets qui ne sont pas encore validés, Après validation , l'état du projet change de *En cours de validation* en *Validé*.

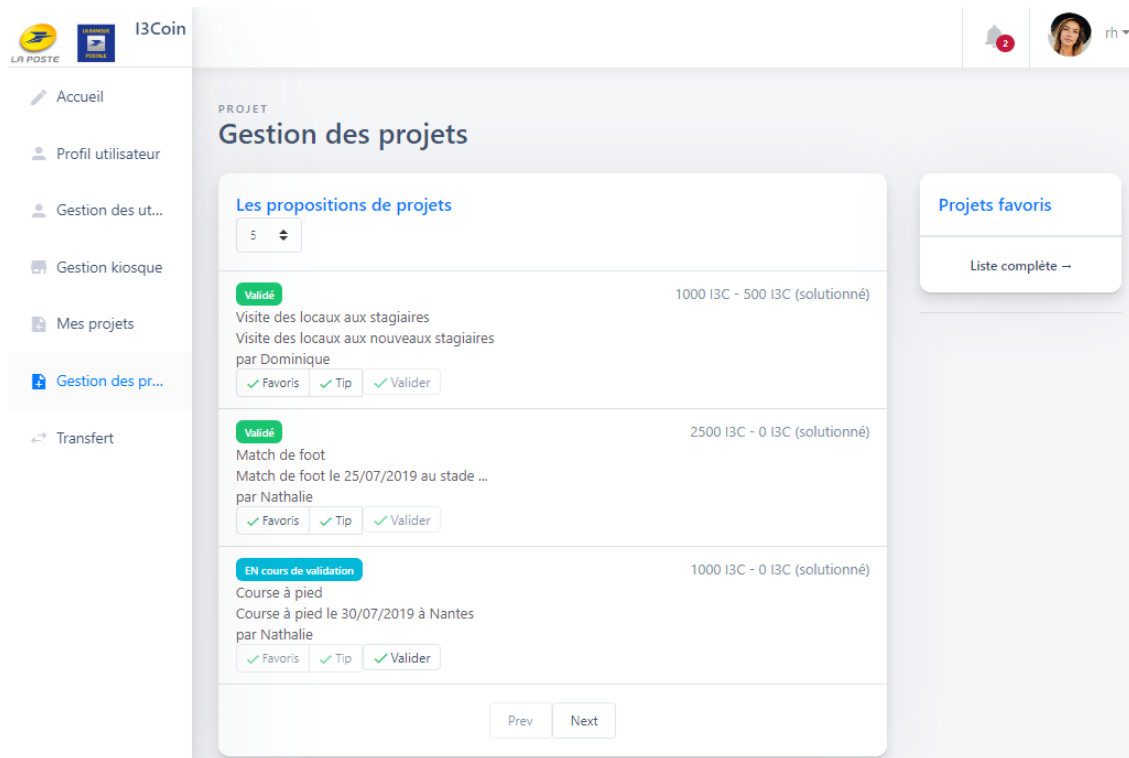


FIGURE 6.23 – page - valider projet

Mettre un projet en favoris

Un utilisateur pourra également cliquer sur “Favoris” pour ajouter ce projet à sa liste de ses projets favoris (pour indiquer qu’il veut participer au projet). La liste des **Projets favoris** est accessible à la partie droite de la page dans le tableau **Projets favoris** en cliquant sur le lien **Liste complète**.

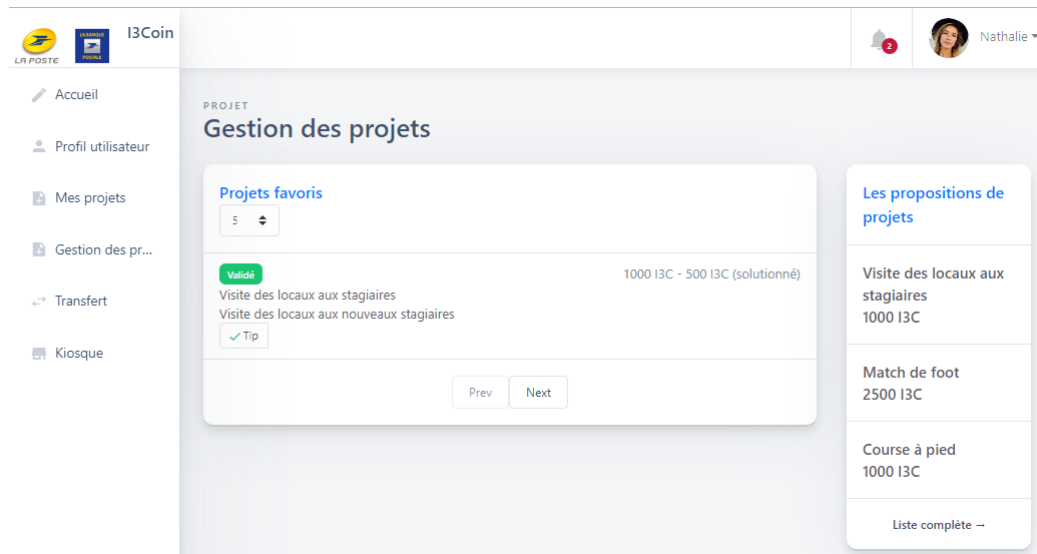


FIGURE 6.24 – page - liste des favoris

Pour retourner sur la liste de tous les projets, il suffit de se rendre à la partie droite droite de la page, dans le tableau **Les propositions de projets** en cliquant sur le lien **Liste complète**.

6.2.2.8 Gestion kiosque (Uniquement visible pour le service RH)

Sur cet écran s’affiche la liste des éléments que l’entreprise propose et qui sont en vente. Pour chaque élément est affiché : le libellé du produit et son prix en i3C. La liste des éléments vendus sont accessibles à la partie droite de la page dans le tableau **Mes ventes**.

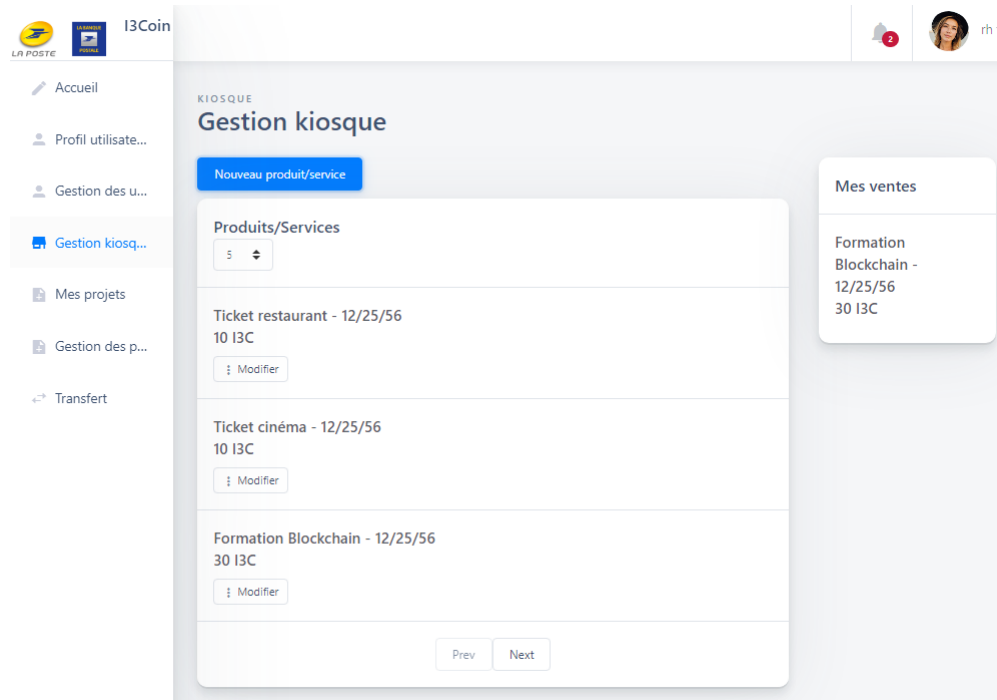


FIGURE 6.25 – page - vente des produits/services (kiosque)

Pour ajouter un produit/service, l'utilisateur doit cliquer sur le bouton "**nouveau produit/service**" et renseigner les champs du formulaire suivants :

- Nom : le nom du produit/ service
- Prix : le coût associé au produit/service
- Description : la description du produit/service

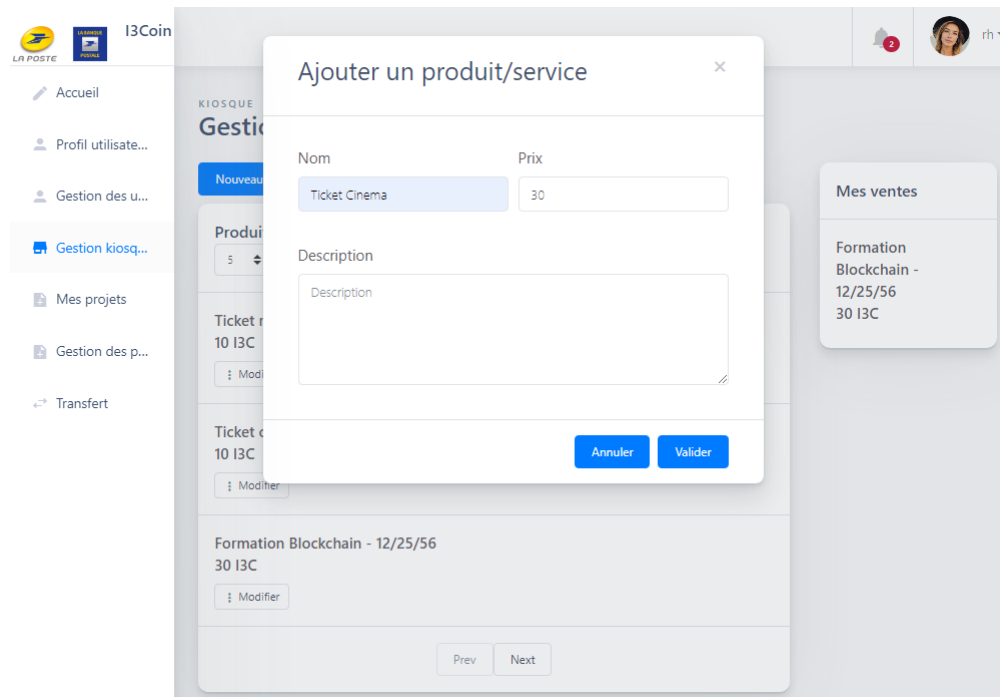


FIGURE 6.26 – page - ajouter des produits/services (kiosque)

6.2.2.9 Kiosque

Sur cet écran s’affiche la liste des éléments que l’utilisateur peut acheter avec son solde de coins i3C. Pour chaque élément est affiché : le libellé du produit et son prix en i3C. Pour effectuer un achat, l’utilisateur doit cliquer sur le bouton **Acheter** d’un élément de la liste. Une popup de confirmation s’affiche alors à l’écran. L’utilisateur peut soit ‘Annuler’ pour revenir à la liste ou ‘Confirmer’ pour confirmer son achat.

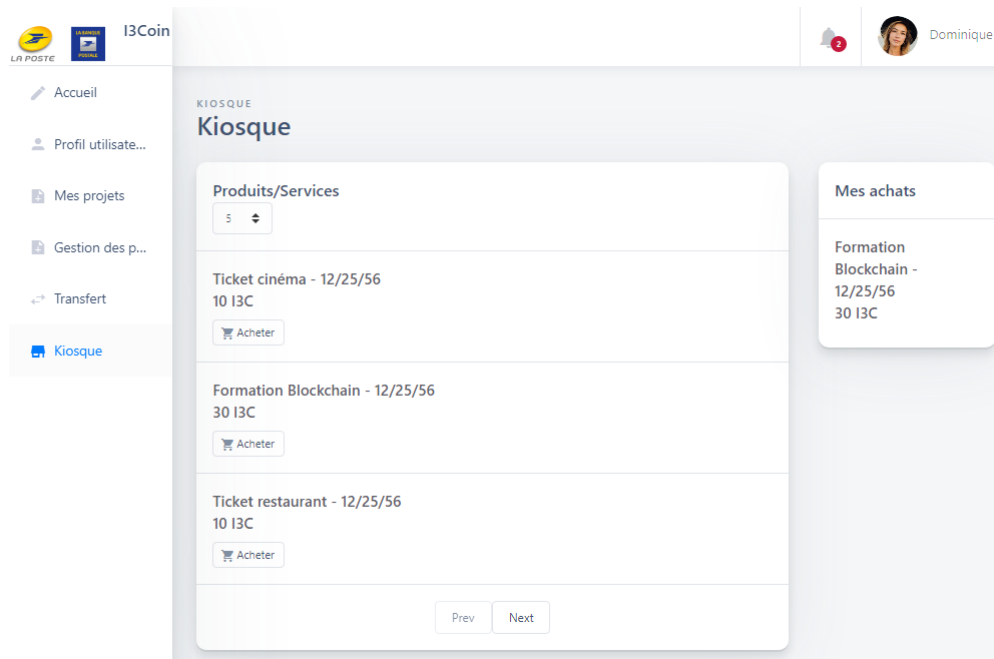


FIGURE 6.27 – page - achat des produits/services (kiosque)

On peut voir à droite le tableau **Mes achats** contenant la liste des produits et services achetés par l'utilisateur connecté.

6.2.2.10 Transfert

Cet écran affiche la liste des transferts entrants et sortants du compte connecté. On peut voir les informations telles que :

- La clé publique du destinataire (s'il s'agit d'un transfert sortant), et la source (s'il s'agit d'un transfert entrant)
- Le montant de la transaction

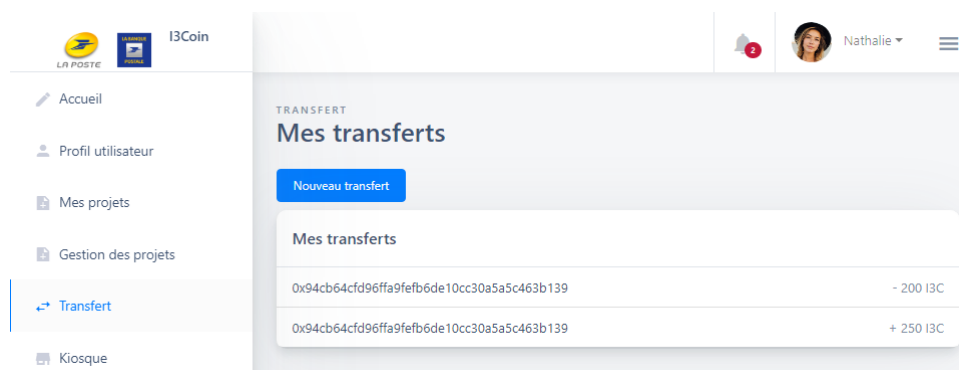


FIGURE 6.28 – page - liste des transferts I3C effectués

Pour effectuer un nouveau transfert, l'utilisateur doit cliquer sur le bouton "**Nouveau transfert**", renseigner les champs du formulaire ci dessous et cliquer sur **Valider** :

- Adresse du destinataire
- montant à transférer
- Le motif du transfert

The screenshot displays the I3Coin web application interface. On the left is a sidebar menu with options: Accueil, Profil utilisat..., Mes projets, Gestion des ..., Transfert (highlighted), and Kiosque. The main content area shows a 'Mes transferts' section with a 'Nouveau' button. A modal window titled 'Affectuer un transfert' is open, containing the following fields:

- Adresse du destinataire**: A text input field containing the hexadecimal address '0x94cb64cfd96ffa9fefb6de10cc30:'.
- Montant**: A text input field containing the value '300'.
- Motif**: A large text area containing the word 'Aucun'.

At the bottom of the modal are two buttons: 'Annuler' and 'Valider'. In the background, a list of transactions is visible, including one for '- 200 I3C' and another for '+ 250 I3C'. The top right corner shows a user profile for 'Nathalie'.

FIGURE 6.29 – page - transférer des I3C

Conclusion et Perspectives

7.1 Conclusion générale

Nous avons donc mis en place une application décentralisée de bout en bout en suivant les étapes suivantes :

- créer une blockchain privée
- écrire le contrat : qui est la base applicative de notre projet, et qui par la suite est distribuée dans la blockchain ;
- déployer le contrat dans notre « blockchain privée »
- écrire l'interface web du projet

Une application décentralisée (aussi appelée dApp) est généralement accessible depuis un site web. Rien de très nouveau ici direz-vous, c'est ce qui se fait dans le web depuis son commencement. Ce qui est nouveau en revanche, c'est que cette application web se connecte non pas à un serveur centralisée (serveur web) mais directement à la blockchain d'Ethereum.

Autrement dit, sur une application web classique (moderne) :

- l'utilisateur se connecte à un serveur web qui lui renvoie l'interface à afficher dans le navigateur (partie front-end) ;
- l'interface se connecte à une API (partie back-end) en lui envoyant des données, qui lui renvoient à son tour d'autres données.

L'application fonctionne en mode centralisé, dans le sens où a priori tous les utilisateurs se connectent au même serveur web, et font donc confiance à ce serveur. Impossible a priori de vérifier que le serveur web centralisé traite les données comme cela est promis. De plus si le serveur tombe (piratage, fermeture de la société...), toutes les données sont potentiellement perdues.

Dans le cas d'une application décentralisée en revanche :

- l'utilisateur se connecte à un serveur web qui lui renvoie l'interface à afficher dans le navigateur (partie front-end, jusque là pas de changement) ;

- l'interface se connecte au portefeuille local de l'utilisateur via une API (web3, nous verrons cela plus tard) en créant un contrat ou en appelant une méthode d'un contrat spécifique;
- le portefeuille exécute le contrat dans la blockchain, puis renvoie le résultat au front-end.

7.2 Perspectives

- Nous avons créé une blockchain privée, mais une blockchain d'un nœud n'est pas très utile. Il faut donc connecter d'autres nœuds à notre blockchain.
- On envisage également réalisé la version mobile de l'application
- Proposer iCubeCoin pour l'application **Ma French Bank**, comme une nouvelle fonctionnalité pour attirer les jeunes.

Bibliographie

- [1] <https://docplayer.fr/81211984-Livret-d-accueil-direction-du-controle-de-gestion-branche-services-courrier-colis-direction-financiere-direction-du-controle-de-gestion.html>
- [2] <https://blockchainfrance.net/decouvrir-la-blockchain/c-est-quoi-la-blockchain/>
- [3] <https://www.ibm.com/blogs/cloud-computing/2017/04/11/characteristics-blockchain/>
- [4] <https://blockchainfrance.net/decouvrir-la-blockchain/c-est-quoi-la-blockchain/>
- [5] <https://bitcoin.org/fr/>
- [6] <https://www.ethereum.org/>
- [7] <https://www.ripple.com/>
- [8] <https://www.bitcoincash.org/>
- [9] <https://litecoin.org/fr/>
- [10] <https://github.com/bitcoin/bitcoin>
- [11] <https://bitcoinexchangeguide.com/>
- [12] <https://geth.ethereum.org/downloads/>
- [13] <https://truffleframework.com/ganache>
- [14] <https://github.com/trufflesuite/ganache/releases/tag/v1.3.1>
- [15] <https://nodejs.org>
- [16] <https://github.com/atom/atom>
- [17] <https://git-scm.com/download/win>
- [18] <https://chrome.google.com/webstore/detail/react-developer-tools/fmkadmapgofadopljbjfkapdkoienihi?hl=en>
- [19] <https://github.com/DesignRevision/shards-dashboard-react>
- [20] Bitcoin : A Peer-to-Peer Electronic Cash System Satoshi Nakamoto October 31, 2008

Annexe **A**

Quelques fonctions du smart contract

A.1

```
function ajouterUtilisateur (string _mail, string _nom,
address _addr) public {
    utilisateurCounter++;
    Utilisateur memory nouvelUtilisateur;
    nouvelUtilisateur.mail=_mail;
    nouvelUtilisateur.nom=_nom;
    nouvelUtilisateur.actif=0;
    nouvelUtilisateur.profil=0;
    nouvelUtilisateur.id=utilisateurCounter;
    listeUtilisateur[_addr] = nouvelUtilisateur;
}

function creerProjet(string _name, string _description, uint256 _price) public {
    projetCounter++;
    operationCounter++;
    Projet memory p;
    p.id = projetCounter;
    p.name=_name;
    p.description=_description;
    p.price = _price;
    p.createur = msg.sender;
    p.etat = 0;
    p.priceSolution = 0;
    projets[projetCounter] = p;
    LogCreerProjet(projetCounter, msg.sender, _name, _price);
}

function tipProjet(uint _id,uint256 _montant) public {
    require(projetCounter > 0);
    require(_id > 0 && _id <= projetCounter);
    Projet storage projet = projets[_id];
    balances[msg.sender]-=_montant;
    balances[projet.createur]+=_montant;
    projet.price+=_montant;
    LogTip(operationCounter,msg.sender,_id,projet.name,_montant,projet.createur);
}
```

FIGURE A.1 – Annexe- fonctions du smart contract (a)

```

function ajouterKiosque(string _name, string _description, uint256 _price) public {
    kiosqueCounter++;

    Kiosque memory p;
    p.id = kiosqueCounter;
    p.name=_name;
    p.description=_description;
    p.price = _price;
    p.etat = 0;
    p.createur=msg.sender;
    kiosques[kiosqueCounter] = p;
}

function updateKiosque(uint _id,string _nom,uint256 _prix, string _description) public{
    require(kiosqueCounter > 0);
    require(_id > 0 && _id <= kiosqueCounter);
    Kiosque storage kiosque = kiosques[_id];
    kiosque.price=_prix;
    kiosque.name=_nom;
    kiosque.description=_description;
    kiosque.createur=msg.sender;
    kiosques[_id]=kiosque;
}

function validerProjet(uint _id) public {

    require(projetCounter > 0);
    require(_id > 0 && _id <= projetCounter);
    Projet storage projet = projets[_id];
    require(projet.etat!=1);
    balances[msg.sender]-=(projet.price+montantPrimeCreationProjet);
    balances[projet.createur]+=(projet.price+montantPrimeCreationProjet);
    projet.etat=1;
    LogValiderProjet(_id, projet.createur, projet.name, projet.price,
        montantPrimeCreationProjet,msg.sender);
}

function solutionnerProjet(uint _id,address _solutionneur,uint256 montant_) public {
    require(projetCounter > 0);
    require(_id > 0 && _id <= projetCounter);
    Projet storage projet = projets[_id];
    require(montant_<=projet.price);

    balances[msg.sender]-=montant_;
    balances[_solutionneur]+=montant_;
    projet.priceSolution+=montant_;
    LogSolutionnerProjet(operationCounter,_solutionneur,_id,projet.name,
        montant_,projet.createur);
}

function transfert(address _receveur,uint256 _montant) payable public {
    transfer(_receveur,_montant);
    LogTransfert(operationCounter,msg.sender,_receveur,_montant);
}

```

FIGURE A.2 – Annexe- fonctions du smart contract (b)

