



ESPRIT SCHOOL OF ENGINEERING

www.esprit.tn - E-mail : contact@esprit.tn

Siège Social : 18 rue de l'Usine - Charguia II - 2035 - Tél. : +216 71 941 541 - Fax. : +216 71 941 889

Annexe : 1-2 rue André Ampère - 2083 - Pôle Technologique - El Ghazala - Tél +216 70 250 000 - Fax +216 70 685454

2021-2022

PROJET DE FIN D'ÉTUDES

SPÉCIALITÉ : INFORMATIQUE

Authentification par blockchain : OTENTIX

Réalisé par : Meriem BADER

Encadré par :

Encadrant ESPRIT : Mme. Hella BANI

Encadrants Entreprise : M. Skander EL-FEKIH & M. Anis KTARI

AGILZ

J'autorise l'étudiante **Meriem BADER**, à faire le dépôt de son rapport de stage de fin d'études en vue d'une soutenance.

Madame **Hella BANI**, encadrante académique.

Signature.



J'autorise l'étudiante **Meriem BADER**, à faire le dépôt de son rapport de stage de fin d'études en vue d'une soutenance.

Monsieur **Skander EL-FEKIH**, encadrant professionnel.

Signature.



Dédicaces

Ce projet de fin d'études est dédié à mes chers parents et mes sœurs, qui m'ont toujours poussée et motivée dans mes études. Sans eux, je n'aurais jamais pu atteindre ce niveau professionnel et éducatif.

Je tiens à remercier mon encadrante académique Mme. Hella BANI et mes encadrants professionnel M. Skander EL-FEKIH et M. Anis KTARI, pour leur accompagnement, leur soutien et leur aide tout au long de la période de stage.

Remerciements

Mes chers parents j'en aurais long et beaucoup à dire, mais ce que je ressens le besoin de faire, c'est de vous dire merci. Pour plein de choses, mais pour une en particulier, celle d'avoir toujours cru en moi. Vous étiez avec moi tout au long de mon parcours éducatif et professionnel, je vous remercie pour vos soutiens physique et morale, ainsi que tous mes amis et collègues avec qui j'ai eu l'honneur de travailler et d'étudier avec.

Je n'oublierai jamais le soutiens et l'encouragement de **mes sœurs**, merci de si bien accomplir vos rôles. Merci d'être là quand ça ne va pas. Merci de m'avoir épaulée quand j'en avais besoin. Merci d'avoir apaisée mes pleurs peu importe la situation, vous avez toujours les mots qu'il faut et vous savez reconnaître les moments où j'ai simplement besoin d'une oreille attentive pour m'écouter.

J'exprime ma reconnaissance et remerciements envers tous mes professeurs d'ESPRIT et tout le cadre administratif qui ont été la première raison de la réussite.

J'adresse mes remerciements à **Madame Hella Bani** pour sa pédagogie, et son aide tout au long de ce projet.

Aussi, je remercie **Monsieur Skander EL-Fekih, Monsieur Anis KTARI**, pour leur soutien jusqu'à la dernière minute, pour moi, ça prouve qu'AGILZ n'est pas seulement une entreprise, mais une famille que j'étais chanceuse d'intégrer.

Que les membres du jury trouvent, ici, l'expression de mes remerciements pour l'honneur qu'ils me font en acceptant de juger ce travail.

Sommaire

Dédicaces	1
Remerciements	3
Introduction Générale.....	10
Chapitre 1 : Présentation du contexte général du projet	11
1. Introduction	12
2. Présentation de l'organisme d'accueil.....	12
3. Etude de l'existant.....	12
3.1 Problématique	12
4. Critique de l'existant	13
4.1 Solutions existantes.....	13
4.2 Critique des solutions existantes	14
5. Solution proposée.....	14
6. Gestion de projet et déroulement du travail	15
6.1 La méthode SCRUM.....	15
6.2 Le sprint agile	16
7. Plan de travail.....	16
7.1 Diagramme de Gant	16
Conclusion.....	17
Chapitre 2 : Analyse et spécification des besoins	18
1. Introduction	19
2. Besoins globaux	19
2.1 Acteurs	19
3. Analyse des besoins globaux.....	19
3.1 Les besoins fonctionnels	19
3.2 Les besoins non fonctionnels	20
4. Product backlog du projet	21
5. Diagramme de cas d'utilisation général	22
5.1 Description des acteurs	22

5.2	Présentation du diagramme de cas d'utilisation.....	22
6.	Conclusion.....	24
Chapitre 3 : Sprint 1 : Mise en place de l'environnement de l'application		25
1.	Introduction	26
2.	Product backlog du sprint 1.....	26
3.	Démarche pour la mise en place des fonctionnalités basiques dans le contrat intelligent ..	27
4.	Dictionnaire des terminologies.....	29
4.1	Blockchain	29
4.2	ERC-721	29
4.3	Monnaie cryptographique (Crypto Currency)	29
4.4	Portefeuille cryptographique (Wallet)	29
4.5	Dapps	30
4.6	Décentralisation	30
4.7	Ether	30
4.8	Ethereum	30
4.9	Gas	30
4.10	Contrat intelligent (Smart Contract).....	30
4.11	OpenZeppelin	30
4.12	Mint un NFT.....	30
4.13	Metadata (métadonnée)	31
4.14	Machine virtuelle d'ethereum (EVM).....	31
4.15	Ether.Js	31
4.16	IPFS	31
4.17	Pinata.....	31
4.18	Non-Fungible Token (NFT)	31
5.	Comparaison entre le stockage centralisé et décentralisé	33
6.	Signature des NFT.....	33
6.1	Concept	33
6.2	Hashage.....	34
6.3	Signature	34
6.4	Vérification des signatures.....	35

7. Choix Technique	37
7.1 Technologies frontend.....	37
7.2 Comparaison entre ReactJs et Angular	38
7.3 Technologies backend.....	39
8. Environnement de travail	40
8.1 Environnement matériel.....	40
8.2 Environnement logiciel.....	40
8.2.1 Visual studio code	40
8.2.2 Windows PowerShell	41
8.2.3 Postman	41
8.2.4 GitBash.....	41
9. Architecture physique	41
Conclusion.....	42
Chapitre 4 : Sprint 2 : Développement des fonctionnalités de l'application	43
1. Introduction	44
2. Architectures logiques.....	44
3. Développement du module NFT	45
3.1 Scénario du cas d'utilisation consulter NFT	45
3.2 Diagramme de séquence « Ajouter un NFT »	46
3.3 Architecture Mint NFT.....	48
3.4 Diagramme de séquence « Bruler un NFT »	49
3.5 Diagramme de séquence « Transférer un NFT »	50
3.6 Diagramme de séquence « Générer un QrCode »	52
4. Présentation des interfaces graphiques.....	52
4.1 Interface d'accueil.....	53
4.2 Interface Marketplace	53
4.3 Interface « Buy NFT »	54
4.4 Interface « Profile »	56
4.5 Interface « Créer un NFT »	57
5. Diagramme de déploiement	61
Conclusion.....	62

Chapitre 5 : Sprint 3 :	63
Test et déploiement de l'application	63
1. Introduction	64
2. Test unitaire	64
3. Déploiement	66
3.1 Vercel API	66
3.2 GitHub	66
Conclusion	67
Conclusion générale	68
Netographie	69

Liste des Figures

Figure 1 - Les différents rôles du scrum	16
Figure 2 - Diagramme de Gant.....	17
Figure 3 - Diagramme de cas d'utilisation générale	23
Figure 4 Démarche sprint 1	27
Figure 5- Signature de la ressource téléchargée en PINATA	35
Figure 6- Signature du NFT	35
Figure 7- Ressource stockée en PINATA	36
Figure 8 - Architecture physique Otentix.....	42
Figure 9 - Architecture logique de l'application.....	45
Figure 10 - Diagramme de séquence « Consulter NFT ».....	46
Figure 11 - Diagramme de séquence « ajouter un NFT ».....	47
Figure 12 - Structure du fichier JSON du NFT.....	48
Figure 13 - Architecture Mint NFT.....	49
Figure 14 - Diagramme de séquence « bruler un NFT ».....	50
Figure 15 - Diagramme de séquence « transférer un NFT ».....	51
Figure 16- Diagramme de séquence « Générer un Qrcode »	52
Figure 17 - Interface d'accueil	53
Figure 18 - Interface liste des NFT	54
Figure 19- Interface d'achat d'un NFT	55
Figure 20- Interface de confirmation de la transaction d'achat d'un NFT	56
Figure 21- Interface profil utilisateur	57
Figure 22- Signature de la ressource téléchargée.....	58
Figure 23- Succès de téléchargement de la ressource signée.....	58
Figure 24 - Signature du NFT	59
Figure 25 - Interface "Fixer un prix pour un NFT"	60
Figure 26- Validation du prix du NFT	60
Figure 27- Modification du prix d'un NFT existant	61
Figure 28 - Diagramme de déploiement.....	62
Figure 29 - Cycle de développement du contrat intelligent.	64
Figure 30 – Test.....	65
Figure 31- Dashbord Vercel & déploiement Otentix	66
Figure 32- Domaines Otentix	67

Liste des tableaux

Tableau 1- Product Backlog générale du projet OTENTIX.....	21
Tableau 2 - Product backlog du sprint 1	26
Tableau 3 - Propriété du NFT	32
Tableau 4- Tableau comparatif entre ReactJs et Angular	39
Tableau 5 - Environnement matériel	40
Tableau 6 - Product backlog du sprint 2 : Développement des fonctionnalités de l'application OTENTIX.....	44

Introduction Générale

Le plus grand obstacle à la migration de nombreux services en ligne est la capacité à sécuriser les données et vérifier l'identité des utilisateurs de ce service. Actuellement, l'authentification en ligne repose sur un mot de passe ou, en de rares occasions, l'utilisation de l'authentification à double facteur. Le problème avec ces méthodes est que les mots de passe sont notoirement non sécurisés et que l'authentification à double repose généralement sur l'envoi d'un code par SMS ou par un service tiers. Une solution à ce problème pourrait être la blockchain. Cette technologie offre de hauts standards de transparence et de sécurité car elle fonctionne sans organe centrale de contrôle.

Avec ces propriétés intrinsèques, il devient évident que la blockchain peut être utilisée comme fournisseur d'identité. Des blocs pourraient stocker les informations. L'authentification à des services pourraient alors s'appuyer sur ces informations. Elle deviendrait alors une transaction sur le réseau de la Blockchain. Grâce aux propriétés de la blockchain, la preuve d'authentification est infalsifiable et tracée de manière pérenne. Le mécanisme d'authentification est donc robuste et potentiellement inviolable.

C'est dans ce cadre que s'inscrit mon stage de fin d'étude à ESPRIT, réalisé à AGILZ, un cabinet de conseil en informatique. Ma tâche consiste à développer une solution d'authentification par blockchain « Otentix ». Dans ce rapport je vais décrire par détails toutes les étapes du travail réalisé, de la conception jusqu'à la réalisation, et qui se compose de 6 chapitres :

Le premier chapitre est consacré à la présentation de l'entreprise accueillante et du contexte général du projet. Nous présentons aussi notre méthode de travail. Le deuxième chapitre présente l'analyse et les spécifications des besoins. Le troisième chapitre présente le premier sprint consacré à la mise en place de l'environnement de l'application. Le quatrième chapitre consacré au développement des fonctionnalités avancées présente la réalisation de chacun des modules développés. Et en fin le cinquième chapitre présente la phase du test et de déploiement de l'application.

Le rapport est clôturé par une conclusion générale qui présente un récapitulatif de ce qui a été réalisé ainsi que les perspectives.

Chapitre 1 : Présentation du contexte général du projet

1. Introduction

Dans le cadre de ce premier chapitre nous présentons, en premier lieu, l'organisme d'accueil. En second lieu, nous mettons l'accent sur l'importance de la solution proposée à travers une analyse et une critique de l'existant.

2. Présentation de l'organisme d'accueil

Lancé en France en 2019 comme un cabinet de conseil en informatique, née de l'association de plusieurs ESN françaises, spécialisées en finance des marchés et asset management. Depuis cette date le métier d'AGILZ est d'aider les institutions du monde de la finance, banque de financement et d'investissement, asset manager et autres acteurs financiers.

Le métier d'AGILZ est aussi de mettre à disposition de ces clients des compétences dans le conseil en organisation, dans la maîtrise d'ouvrage et la maîtrise d'œuvre, afin de répondre au mieux aux problématiques front et back.

3. Etude de l'existant

Dans cette section, nous présentons la problématique, les solutions existantes, une critique des solutions existantes et la solution proposée.

3.1 Problématique

Les identifiants sont le principal vecteur de failles de sécurité avec 65% d'incidents liés au vol d'identités. Les conséquences pour l'entreprise sont souvent désastreuses, surtout avec l'entrée en vigueur du RGPD¹, et les sanctions financières lourdes (jusqu'à 4% du chiffre d'affaires ou 20 millions d'euros). Avec 214 données piratées toutes les secondes, il semble urgent de revoir les politiques d'authentification et d'identification client.

Sur le banc des accusés on retrouve toujours les fameuses authentification et identification par email/mot de passe. Ceci impose donc aux marques et organisations d'entreprendre les démarches essentielles pour rassurer leurs clients, tout en garantissant une expérience fiable.

Les solutions qui existent et qui sont organisées autour des technologies de Customer Identity and Access Management (CIAM²) concernent plutôt la problématique d'uniformisation des accès et des données client pour les outils existant au sein de la même entreprise.

¹ RGPD : Le règlement Générale sur la Protection des Données

² CIAM : Gestion des accès et identité Clients

Il est impératif de repenser la gestion des identités et des authentifications pour les applications qui sont engagées dans des opérations de transformation digitale et de mise en conformité avec les règlements sur la protection des données.

Alors, comment conjuguer authentification, sécurité et expérience client ?

4. Critique de l'existant

4.1 Solutions existantes

Il existe plusieurs solutions pour l'authentification qui permettent de s'assurer que la personne qui se connecte au réseau est bien celle qu'elle prétend être.

Parmi les solutions les plus utilisées sur le marché nous pouvons citer :

- **Authentification par mot de passe :** Quiconque utilise Internet connaît les mots de passe, la forme d'authentification la plus élémentaire. Une fois qu'un utilisateur a saisi son nom d'utilisateur, il doit saisir un code secret pour accéder au réseau. Si chaque utilisateur garde son mot de passe privé, selon la théorie, tout accès non autorisé sera empêché. Cependant, l'expérience a montré que même les mots de passe secrets sont vulnérables au piratage. Les cybercriminels utilisent des programmes qui essaient des milliers de mots de passe potentiels, obtenant l'accès lorsqu'ils devinent le bon.
- **Authentification à deux facteurs (2FA) :** L'authentification à deux facteurs s'appuie sur les mots de passe pour créer une solution de sécurité beaucoup plus robuste. Il faut à la fois un mot de passe et la possession d'un objet physique spécifique pour accéder à un réseau, quelque chose que vous connaissez et que vous possédez. Les guichets automatiques ont été l'un des premiers systèmes à utiliser l'authentification à deux facteurs. Pour utiliser un guichet automatique, les clients doivent mémoriser un « mot de passe » (leur code PIN) et insérer une carte de débit. Ni l'un ni l'autre ne se suffit à lui-même. En sécurité informatique, 2FA suit le même principe. Après avoir saisi leur nom d'utilisateur et un mot de passe, les utilisateurs doivent franchir un obstacle supplémentaire pour se connecter : ils doivent saisir un code à usage unique à partir d'un appareil physique particulier. Le code peut être envoyé sur leur téléphone portable par SMS ou il peut être généré à l'aide d'une application mobile. Si un pirate informatique devine le mot de passe, il ne peut pas continuer sans le téléphone portable de l'utilisateur à l'inverse, s'ils volent l'appareil mobile, ils ne peuvent toujours pas entrer sans le mot de passe. 2FA est mis en œuvre sur un nombre croissant de sites Web bancaires, de messagerie électronique et de médias sociaux. Chaque fois que c'est une option, assurez-vous de l'activer pour une meilleure sécurité.
- **Authentification par transaction :** L'authentification des transactions adopte une approche différente des autres méthodes d'authentification Web. Plutôt que de s'appuyer

sur les informations fournies par l'utilisateur, il compare plutôt les caractéristiques de l'utilisateur avec ce qu'il sait de l'utilisateur, à la recherche de divergences. Par exemple, disons qu'une plateforme de vente en ligne a un client avec une adresse personnelle au Canada. Lorsque l'utilisateur se connecte, un système d'authentification des transactions vérifie l'adresse IP de l'utilisateur pour voir si elle correspond à son emplacement connu. Si le client utilise une adresse IP au Canada, tout va bien. Mais s'il utilise une adresse IP en Chine l'application va être déverrouillée. Ce dernier cas déclenche des étapes de vérification supplémentaires. L'utilisateur réel peut simplement voyager en Chine, donc un système d'authentification des transactions devrait éviter de le verrouiller complètement. L'authentification des transactions ne remplace pas les systèmes basés sur un mot de passe, au lieu de cela, il fournit une couche de protection supplémentaire.

L'authentification par mot de passe est la méthode de protection la plus courante. Les mots de passe peuvent être sous la forme d'une chaîne de lettres, de chiffres ou de caractères spéciaux. Pour vous protéger, vous devez créer des mots de passe forts qui incluent une combinaison de toutes les options possibles.

Cependant, les mots de passe sont sujets aux attaques de phishing et à une mauvaise hygiène qui affaiblit leur efficacité. Une personne moyenne possède environ 25 comptes en ligne différents, mais seulement 54 % des utilisateurs utilisent des mots de passe différents sur leurs comptes.

4.2 Critique des solutions existantes

L'inconvénient majeur de la solution existante est que l'utilisateur ne peut pas s'authentifier sans utiliser ses données personnelles, ce qui entraîne beaucoup de problèmes :

- Les informations personnelles ne sont pas en sécurité si elles sont gardées secrètes.
- Manque de contrôle d'identité.
- Les données personnelles peuvent être falsifier par une autre personne.

5. Solution proposée

L'authentification est nécessaire pour identifier en toute sécurité les utilisateurs en ligne. Après une étude approfondie de l'existant, nous proposons la conception et le développement d'une solution d'authentification par blockchain. Notre solution permet à l'utilisateur de s'authentifier à travers un jeton non fongible (³NFT, de l'anglais non-fungible token) un jeton cryptographique qui représente un objet auquel est rattachée une identité numérique qui est reliée à un propriétaire. Dans ce NFT. Grâce au mécanisme de clef publique clef privée, on garantit l'authentification du propriétaire du NFT. La paire clef publique/ clef privée est quant à elle issue du compte crée. Son authentification est validée grâce à un protocole de la blockchain.

³ NFT: Non Fungible Token

6. Gestion de projet et déroulement du travail

Dans le cas de notre projet, nous avons opté pour SCRUM comme méthode de développement informatique, car nous avons jugé qu'elle est la méthode la plus adéquate, puisqu'elle dispose d'une organisation adaptée et d'un état d'esprit Agile, qui privilégie l'esprit d'équipe non seulement dans la réalisation technique, mais aussi pour la participation du client à la réalisation du projet.

La Méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients.

Elle permet aussi de suivre l'évolution du travail par la gestion des problèmes qui vont nous permettre de faire des corrections, permettant ainsi d'entreprendre des actions correctrices sans trop de pénalités dans les coûts et les délais. Il y a plusieurs méthodes AGILE et il ne s'agit pas de choisir la meilleure méthode parmi celles existantes. Pour notre projet, nous nous sommes orientés vers une méthode de type AGILE et plus particulièrement SCRUM.

6.1 La méthode SCRUM

Scrum est une méthode agile utilisée dans le développement de logiciels. Elle vise à satisfaire au mieux les besoins du client tout en maximisant les probabilités de réussite du projet, un projet utilisant Scrum est composé d'une suite d'itérations courtes de l'ordre de 3 à 6 semaines appelées sprints.

A la fin d'un sprint, l'équipe livre au client un incrément de logiciel fini potentiellement livrable. Le projet peut être réorienté par le client à la fin de chaque sprint. Le backlog du produit constitue l'ensemble du travail à un instant t. Le travail à faire durant un sprint est listé dans le backlog de ce dernier.

6.1.1 Les rôles dans SCRUM

— Product Owner:

Les Product Owner qui sont **M. Skander El FEKIH** et **M. Anis KTARI**, ont participé à l'élaboration des besoins participé la vision client du produit final.

— Scrum master :

Les Scrum Master, **M. Skander EL FEKIH** et **M. Anis KTARI**, qu'il ne faut absolument pas confondre avec des chefs de projet, ont pour rôle d'animer et de faciliter le travail de l'équipe de développement.

— Equipe :

L'équipe de développement qui se compose de moi-même et mes chefs de stage, **M. Skander EL FEKIH** et **M. Anis KTARI**.



Figure 1 - Les différents rôles du scrum

6.2 Le sprint agile

Le sprint agile représente le cœur de la méthode scrum. Cette qualification lui correspond plutôt bien, puisque tous les développements incrémentaux menant petit à petit au produit final du projet sont réalisés au sein des sprints, ce sprint permet de faire un daily meeting pour voir les points manquants et le progrès du travail.

7. Plan de travail

7.1 Diagramme de Gant

La figure ci-dessous présente le diagramme de Gant qui consiste un ensemble de mesures et activités à entreprendre pour réaliser un projet. La durée développement de notre projet OTENTIX va s'étaler sur 6 mois, nous avons cette durée en deux trimestres. Le premier trimestre a été

consacré à la recherche et à la réalisation du projet. Alors que, le second semestre, nous l'avons consacré à la réalisation, le test et le déploiement de l'application.

Désignations	1er trimestre 2022			2ème trimestre 2022		
	Jan 22	Fev 22	Mars 22	Avril 22	Mai 22	Juin 22
Documentation, Recherche						
Réalisation, Recherche						
Réalisation, Test						
Test,						
Mise en production						

Figure 2 - Diagramme de Gant

Conclusion

Ce premier chapitre constitue une étape primordiale pour fixer les repères de notre projet. Après avoir présenté l'organisme d'accueil et avoir identifié les attentes du projet, nous avons déterminé le cadre du travail ainsi que la méthodologie à emprunter lors de ce stage. Dans le cadre du second chapitre, nous analysons et spécifions les besoins fonctionnels et non fonctionnels.

Chapitre 2 : Analyse et spécification des besoins

1.Introduction

Dans le cadre de ce premier chapitre, nous présentons la spécification ainsi que la conception de notre système, ce qui nous amène à identifier les différents besoins fonctionnels et non fonctionnels de notre solution et à présenter le diagramme de cas d'utilisation générale.

2. Besoins globaux

Dans cette partie on va mentionner les différentes fonctionnalités du projet, ainsi que les différents acteurs.

2.1 Acteurs

Avant citer les différentes fonctionnalités, on doit tout d'abord identifier les acteurs de notre projet :

Propriétaire : Le propriétaire a les droits administrateurs, c'est celui qui est le responsable de la gestion des fonctionnalités.

Utilisateur simple : c'est un simple utilisateur qui va utiliser les fonctionnalités qui seront offertes par l'application.

3. Analyse des besoins globaux

Les besoins sont divisés en deux catégories, des besoins fonctionnels et non fonctionnels.

3.1 Les besoins fonctionnels

Les besoins fonctionnels sont l'expression de ce que le produit ou le service délivré par le projet devrait être ou faire, ainsi on va lister les besoins selon l'acteur.

On va lister les besoins selon l'acteur.

Le projet doit permettre au propriétaire de :

- Créer un NFT.
- Signer un NFT.
- Télécharger une ressource en PINATA.
- Signer une ressource.
- Consulter son profil.
- Lister ces NFT.
- Publier un NFT.
- Générer un QrCode.
- Bruler un NFT.

- Modifier un NFT.
- Préciser le prix d'un NFT.

Le projet doit permettre à un utilisateur simple de :

- Consulter la liste des NFT publié.
- Acheter un NFT.
- Consulter la liste de ces NFT.
- Publier un NFT en marketplace.
- Scanner un QrCode d'un NFT.

3.2 Les besoins non fonctionnels

Les exigences non fonctionnelles déterminent les limitations et les contraintes qui peuvent peser sur notre système. Parmi les besoins non fonctionnels de notre application nous citons :

— Maintenabilité

Le code de l'application était bien lisible et compréhensible pour pouvoir le maintenir facilement et rapidement, il est piloté par la méthodologie TDD (test driven developement).

— Performance

- Temps de réponse - chargement de l'application, temps d'ouverture et de rafraîchissement de l'écran seront dans les normes mondiales.
- Délais de traitement - fonctions, importations seront dans les normes mondiales.
- Temps de requête et de rapport - charges initiales et charges suivantes respectant les normes.

— Fiabilité

En près de huit ans, aucune attaque n'a affecté le protocole de la blockchain elle-même, seulement les applications qui y sont attachées. Du moins concernant les grandes blockchains publiques comme Ethereum qu'on a utilisées. Ces protocoles étant répandus dans un grand nombre de serveurs un peu partout sur la planète et dupliqués dans leur intégralité, les attaquer coûterait plus cher que de contribuer à leur évolution.

— Utilisabilité

- L'ensemble des pages de l'application doit être accessible en 3 clics maximum depuis la page d'accueil.
- L'application doit donner une image moderne de l'entreprise

— Compatibilité

- Compatibilité sur différents systèmes d'exploitation.
- Compatibilité sur différentes plateformes.

4. Product backlog du projet

Après avoir cité les besoins fonctionnels et non fonctionnels, nous décrivons le product backlog avec les priorités.

Le product backlog est divisé selon les sprints, chaque sprint comporte son propre product backlog, ce qui a rendu le travail plus facile et nous a donné la main à bien éliminer les problèmes qui vont nous rencontrer au cours du développement du projet, de choisir lesquelles des fonctionnalités on doit commencer avec, des plus importantes au moins importantes.

Le tableau ci-dessous présente le product backlog du projet.

ID	User Stories	Priorité
1	En tant qu'utilisateur, je peux me connecter et créer mon profil	Moyenne
2	En tant qu'utilisateur, je peux consulter la liste des NFT	Faible
4	En tant que propriétaire, Je peux gérer les NFT, bruler, modifier.	Fort
5	En tant que propriétaire Je peux signer un NFT	Fort
6	En tant que propriétaire, Je peux lister les NFTs	Moyenne
7	En tant que propriétaire, Je peux minter des NFTs	Fort
8	En tant que propriétaire, Je peux transférer un NFT	Fort
9	En tant qu'administrateur, Je peux générer un QrCode	Faible

Tableau 1- Product Backlog générale du projet OTENTIX

5. Diagramme de cas d'utilisation général

Dans cette section, nous présentons la description des acteurs et les diagrammes de cas d'utilisation de notre application.

5.1 Description des acteurs

Dans notre application nous avons deux acteurs : un propriétaire (Owner) et un utilisateur simple. Le rôle de chaque acteur est détaillé ci-dessous :

- **Propriétaire ou Owner :** Cet acteur a tous les droits d'accès aux fonctionnalités du système. Il peut administrer des NFT, les signer, transférer un NFT et consulter les différentes interfaces de l'application.
- **Utilisateur simple :** Cet acteur peut consulter la liste des NFT et acheter un NFT.

Chacun de ces acteurs doit s'authentifier obligatoirement à travers son portefeuille pour accéder à l'interface du système.

5.2 Présentation du diagramme de cas d'utilisation

Dans cette section, nous présentons le diagramme de cas d'utilisation général, puis des diagrammes détaillant quelques cas d'utilisation.

La figure 2 présente le diagramme de cas d'utilisation général de l'application qui illustre les différentes fonctionnalités de notre application :

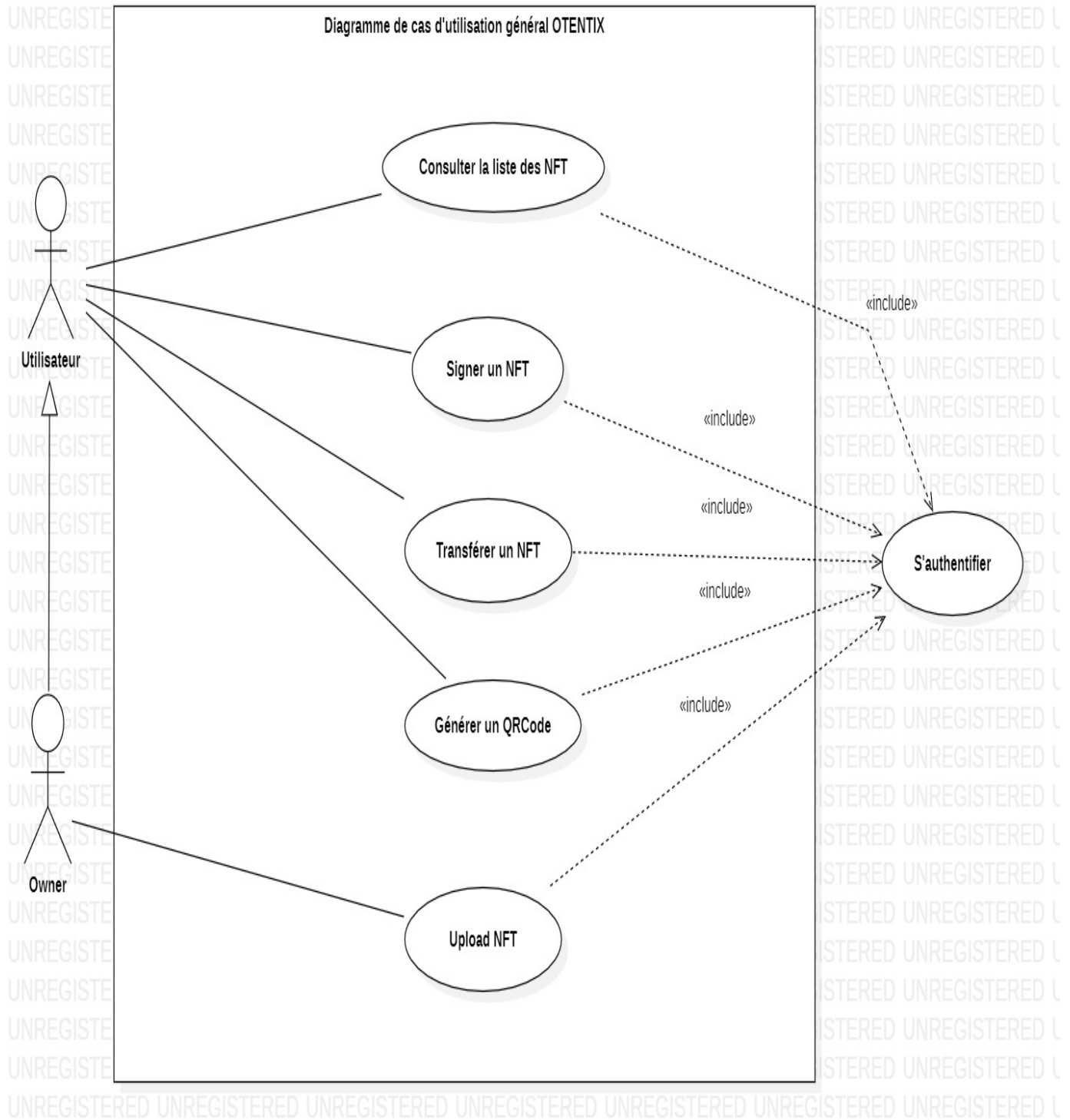


Figure 3 - Diagramme de cas d'utilisation générale

Notre diagramme de cas d'utilisation général englobe les fonctionnalités principales de l'application, il comprend six cas d'utilisation, et deux types d'utilisateurs un propriétaire et un utilisateur simple. On ne peut pas y accéder sans s'authentifier.

Un utilisateur simple peut faire la consultation de la liste des NFT dans la marketplace, peut acheter un NFT et devenir le propriétaire du NFT en question. En plus des fonctionnalités auxquelles un utilisateur simple a accès, le propriétaire peut créer un NFT, le signer, peut mettre un NFT, publier un NFT dans la marketplace, il peut aussi générer un QRCode pour chaque NFT, modifier le prix d'un NFT déjà publié et préciser le prix de chaque NFT lors de la création.

Le travail va être divisé en 3 sprints, chaque sprint contient un ensemble de tâches à accomplir.

Le sprint 1 : Sera consacré à fonctionnalités basiques dans le contrat intelligent et la connexion avec la blockchain dans l'application.

Le sprint 2 : Sera consacré au développement des différentes fonctionnalités mentionnées ci-dessus.

Le sprint 3 : Sera dédié au test et au déploiement de l'application.

6. Conclusion

Dans ce chapitre nous avons présenté les besoins fonctionnels et non fonctionnels, ensuite, nous avons décrit les différentes fonctionnalités du projet sous forme du product backlog, ainsi que le diagramme de cas d'utilisation général de l'application.

Le chapitre suivant présente le premier sprint qui est la mise en place des fonctionnalités basiques dans le contrat intelligent et la connexion avec la blockchain.

Chapitre 3 : Sprint 1 : Mise en place de l'environnement de l'application

1.Introduction

Après avoir analysé les besoins globaux, nous détaillons les différentes étapes faites dans le premier sprint qui est la mise en place des fonctionnalités basiques dans le contrat intelligent avec la connexion avec la blockchain.

2. Product backlog du sprint 1

Dans un premier temps, nous présentons le product backlog du premier sprint, qui va détailler les différentes étapes à suivre par priorité.

ID	Feature	Priorité
1	La création du contrat intelligent en le connectant avec truffle et ganache	Elevée
2	La mise en place des fonctions pour le mint des NFT.	Elevée
3	Le développement des tests.	Moyenne
4	La connexion du contrat intelligent avec le IPFS.	Moyenne
5	La connexion du contrat intelligent avec la partie react.	Elevée
6	Lister les NFT dans la partie frontEnd.	Moyenne
7	Connecter avec le Web3 provider (metamask).	Elevée
8	Développement des hooks pour le portefeuille metamask	Elevée

Tableau 2 - Product backlog du sprint 1

3. Démarche pour la mise en place des fonctionnalités basiques dans le contrat intelligent

Dans n'importe quelle application, il y'a une démarche à suivre pour construire des bases solides au projet pour enfin commencer à développer.

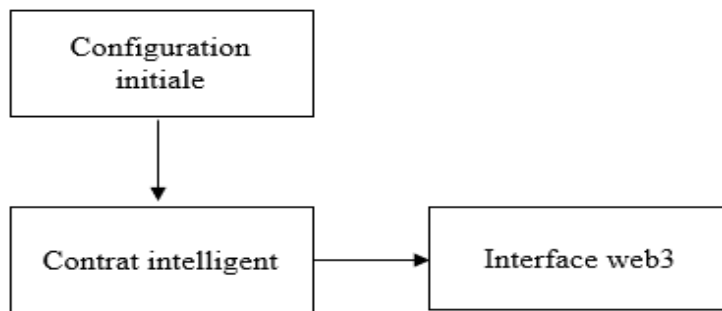


Figure 4 Démarche sprint 1

Configuration initiale :

Cette étape consiste à créer un projet et le configurer avec un environnement de test locale truffle, pour avoir une blockchain locale contenant des portefeuilles virtuels

- **Téléchargement sur IPFS :**

Les NFT ne stockent pas réellement l'image sur la blockchain. Au lieu de cela, ils stockent un hachage de l'image. Ce hachage est appelé l'ID de contenu (⁴CID) du NFT et est généralement hébergé sur ⁵IPFS.

Une fois le contenu téléchargé, il ne peut pas être modifié sans changer le CID.

- **Configuration du Truffle & ganache :**

⁴ CID : Identifiant de contenu

⁵ IPFS : Interplanetary File System

Truffle est une chaîne d'outils de développement qui permet de configurer et de déployer des contrats intelligents.

- **Test et déploiement du contrat intelligent :**

Après avoir créé et configuré le projet avec le truffle, on doit tester les fonctionnalités existantes par des lignes de commande dans la console.

Contrat intelligent :

Les contrats intelligents sont de simples programmes stockés sur un réseau blockchain. Vous pouvez dire que c'est comme un accord entre deux personnes sous la forme d'un code informatique. Les transactions d'un contrat intelligent sont traitées par la blockchain et stockées sous la forme d'une adresse hexadécimale de 42 caractères avec le préfixe "0x"). Tout cela signifie qu'ils peuvent être envoyés automatiquement sans avoir besoin d'un tiers.

- **Déployer le contrat intelligent :**

Avant de pouvoir créer l'application, nous devons déployer le contrat sur un réseau blockchain en localhost. Puis la compilation à partir d'un terminal.

Interface WEB 3 :

Le terme Web3 fait généralement référence à des applications décentralisées qui utilisent des contrats intelligents Ethereum pour remplacer les serveurs web traditionnels.

Le processus de création d'un contrat intelligent et l'interaction avec lui sur le web à l'aide d'Ether.js et de React et nextJS. L'application peut créer des jetons non fongibles (NFT). Il existe de nombreuses technologies différentes impliquées, mais l'idée de base est d'accéder à l'API d'un contrat intelligent à partir d'une application web frontale.

- **Vérifier le plugin Wallet :**

Avant d'utiliser l'application, l'utilisateur doit avoir installé MetaMask.

- **Obtenir le solde du portefeuille :**

L'application utilise ether.js pour interagir avec le portefeuille de l'utilisateur et la blockchain.

- **Boucler les NFT existants :**

Nous utilisons ether.js pour faire référence au contrat déployée. Nous demandons le nombre total de jetons créés.

- **Minter un nouveau Jeton (NFT) :**

Lorsque le bouton mint est cliqué, il connecte le portefeuille à l'utilisateur au contrat intelligent sur la blockchain, puis mint un nouveau jeton en appelant les méthodes définies dans le code solidity.

4. Dictionnaire des terminologies

Dans le cadre de notre projet, nous ressemblons tous les termes liés aux NFT :

4.1 Blockchain

Blockchain est un grand livre numérique décentralisé qui enregistre chaque transaction de manière à ce qu'elle ne puisse pas être altérée ou modifiée. La blockchain peut être décrite comme une collection d'enregistrements liés les uns aux autres et protégés par cryptographie. Tout le monde sur le réseau blockchain a une copie du grand livre. Ainsi, aucune personne ne peut apporter des modifications au grand livre car tout le monde le signalera comme invalide.

4.2 ERC-721

ERC signifie Ethereum Request for Comment.

ERC-721 est la première interface standardisée pour la création de NFT. Il contient un ensemble de règles pour les NFT les définissant comme un moyen d'identifier quelque chose de manière unique. Il est basé sur le déploiement d'un contrat intelligent distinct pour chaque type ou collection de jetons.

4.3 Monnaie cryptographique (Crypto Currency)

Les crypto-monnaies sont une forme de monnaie numérique ou virtuelle qui fonctionne sur la technologie blockchain. Ils sont insensibles à la contrefaçon. Ils ne nécessitent pas d'autorité centrale et sont protégés par des algorithmes de cryptage puissants et complexes. Bitcoin, Litecoin et Ethereum sont les crypto-monnaies les plus courantes disponibles.

4.4 Portefeuille cryptographique (Wallet)

Logiciel qui détient des clés privées et publiques pour les transactions de crypto-monnaie. Il ne stocke jamais de pièces ou de jetons réels.

Une clé publique : est une adresse de portefeuille sous la forme d'un code cryptographique, qui peut être partagée pour recevoir de la crypto-monnaie.

Une clé privée : Est très confidentielle pour le client, car elle est utilisée pour autoriser toute transaction via votre portefeuille.

Il existe 3 types de portefeuilles crypto :

- a) **Portefeuilles logiciels basés sur Exchange** - portefeuilles intégrés dans un échange centralisé comme Vault et CoinDCX.
- b) **Portefeuilles matériels** - accédez aux avoirs cryptographiques uniquement en connectant physiquement USB à un ordinateur. Par exemple : grand livre, Trezor

- c) **Portefeuilles logiciels basés sur un navigateur** - que nous pouvons contrôler, installer sur notre système et conserver la clé privée. Ils sont la porte d'entrée vers le monde du Web3.0 et des NFT. Par exemple: portefeuilles Metamask, Coinbase et Trust.

4.5 Dapps

Dapps signifie applications décentralisées. Ce sont des applications basées sur des réseaux de contrats intelligents comme les blockchains Ethereum. L'ensemble des opérations est stocké sur une blockchain et est open source afin que d'autres personnes puissent y accéder.

4.6 Décentralisation

La décentralisation est le processus consistant à mettre la confiance entre les mains d'un groupe de personnes au lieu d'une seule entité centralisée et l'ensemble du groupe suivra vos transactions.

4.7 Ether

Ether est le carburant de la blockchain Ethereum. La crypto-monnaie native de la plateforme Ethereum.

4.8 Ethereum

Ethereum est une plate-forme logicielle publique décentralisée ouverte basée sur la blockchain qui facilite les contrats intelligents et les applications décentralisées appelées dapps.

4.9 Gas

Gas fait référence aux frais requis pour mener à bien une transaction sur la blockchain Ethereum. Il est nécessaire d'acheter ou de vendre des NFT, de placer des offres et de transférer des NFT vers un autre portefeuille. Habituellement libellé en GWei.

4.10 Contrat intelligent (Smart Contract)

Les contrats intelligents permettent aux utilisateurs d'échanger des valeurs sans avoir besoin d'intermédiaires. Ce sont des accords avec des termes définis et des protocoles pour les faire respecter. Contrairement aux contrats traditionnels qui sont écrits dans des langages humains, les contrats intelligents sont écrits en code qu'un ordinateur peut exécuter.

4.11 OpenZeppelin

OpenZeppelin est une bibliothèque pour le développement sécurisé de contrats intelligents.

4.12 Mint un NFT

Le mint est le processus consistant à placer un NFT dans la blockchain pour la première fois. Le mint est limité par des frais de transaction.

4.13 Metadata (métadonnée)

Les métadonnées sont essentiellement les informations sur les données. Les métadonnées d'un NFT sont simplement les attributs d'un NFT comme le nom, la description et l'image.

4.14 Machine virtuelle d'ethereum (EVM)

- Environnement d'exécution isolé du réseau
- Logiciel exécuté par des ordinateurs du monde entier
- Exécute les contrats intelligents.

4.15 Ether.Js

- Bibliothèque complète et compact
- Interagit avec la blockchain Ethereum et son écosystème.
- Licence MIT (y compris toutes les dépendances)
- Open Source.
- Sécurisée

4.16 IPFS

- Un système de partage de fichier Peer-To-Peer.
- Décentralisé
- Distribué sur chaque nœud qui devient un serveur.
- Utilise un adressage basé sur le contenu.
- Sécurisé.
- Ne tombe jamais en panne.

4.17 Pinata

- Interface d'IPFS.
- Facilite l'hébergement de fichier sur IPFS.
- Permet de télécharger des fichiers image ainsi que des métadonnées JSON sur pinata.

4.18 Non-Fungible Token (NFT)

Lors de l'étude de l'écosystème de la plate-forme NFT, la première chose à considérer est sa définition. Les codes d'identification et les métadonnées d'origine rendent chacun des NFT différents les uns des autres. Puisqu'il s'agit d'actifs numériques, ils ne peuvent pas être échangés ou négociés à une unité équivalente. Les NFT peuvent se présenter sous diverses formes telles que des photos, de la musique, des vidéos et autres.

Chaque produit NFT doit être unique. De nombreuses fonctionnalités les rendent attrayants, contrairement aux autres monnaies numériques. Voici les propriétés spécifiques des NFT :

Propriété	Explication
Indivisibilité	Il n'est pas possible de diviser les jetons NFT en petite coupures.
Authenticité	Chaque jeton non fongible a un propriétaire, la propriété peut être vérifiée facilement.
Non-interopérabilité	Les NFT ne sont pas égaux, ce qui rend difficile leur échange en tant que crypto-monnaies populaires telles que BTC et ETH.
Négociabilité	On peut échanger des NFT dans un régime non-stop. C'est pourquoi leur liquidité est plutôt élevée. Un large éventail de clients pourrait avoir intérêt à acheter ou à vendre ces jetons NFT.
Liquidité	On peut échanger des NFT dans un régime non-stop. C'est pourquoi leur liquidité est plutôt élevée. Un large éventail de clients pourrait avoir intérêt à acheter ou à vendre ces jetons NFT.
Programmabilité	Une plateforme NFT représente une mécanique compliquée qui implique la forge, l'artisanat et la génération. Cette niche a une quantité infinie de chances et de variations. C'est un grand espace de créativité.
Rareté	Les programmeurs peuvent appliquer diverses caractéristiques particulières qui sont impossibles à modifier une fois lancées.

Tableau 3 - Propriété du NFT

5. Comparaison entre le stockage centralisé et décentralisé

Imaginez que vous téléchargez une image d'un chien (appelée dog.jpeg) vers un service de stockage centralisé. L'image de votre chien serait alors disponible en accédant à une URL (*quelque chose comme* <https://mystorage.com/dog.jpeg>).

Cependant, il est très facile d'échanger cette image contre une autre. Je pourrais télécharger une autre image avec le même nom (dog.jpeg) qui remplace l'image originale.

Maintenant, si j'ai visité la même URL qu'avant (<https://mystorage.com/dog.jpeg>), je verrai une image différente. Les gens dépensent des milliers de dollars en NFT et ils seraient énervés si vous remplaciez simplement un avatar avec des traits extrêmement rares par autre chose.

— Le stockage centralisé peut être supprimé

Supposons que vous téléchargez une image sur Google Drive ou AWS. Si vous supprimez l'image de ces services ou si les services eux-mêmes se ferment, l'URL pointant vers l'image se briserait. Par conséquent, il est très facile de tirer le tapis si vos images et données existent sur un service de stockage centralisé.

Pour ces raisons, presque tous les projets NFT sérieux utilisent un service appelé IPFS (ou Interplanetary File System).

IPFS est un système de partage de fichiers peer-to-peer qui est décentralisé, utilise un adressage basé sur le contenu et est sécurisé.

— IPFS ne tombe jamais en panne

Comme la plupart des systèmes décentralisés (comme les blockchains), IPFS ne tombe jamais en panne. Cela signifie qu'une fois que vous avez téléchargé un fichier (ou une image) sur IPFS, il sera toujours disponible tant qu'au moins un nœud du réseau possède le fichier.

Cela signifie que vous ne pouvez pas tirer le tapis à volonté. Il n'y a pas non plus de menace que le système soit arrêté.

6. Signature des NFT

6.1 Concept

A ce niveau on commence par assurer la signature des NFT par l'administrateur avec des clés afin de prouver l'authenticité.

Cette étape est primordiale pour le déploiement sur la blockchain puisque c'est la signature qui va être stockée dans cette chaîne de bloc.

6.2 Hashage

Le hachage est le processus consistant à prendre toutes les données d'entrée et à les faire passer par un algorithme qui produit ensuite des données de sortie d'une taille spécifique et cohérente.

6.2.1 Keccak

L'algorithme SHA-3 (Secure Hash Algorithm 3) / Keccak est l'un des algorithmes de hachage les plus sûrs et les plus efficaces.

- Rapide.
- Sécurisé.

6.2.2 ECDSA

ECDSA est l'algorithme de signature électronique à clé publique utilisé par ethereum. Il fait appel à la cryptographie sur les courbes elliptiques. La sécurité d'ECDSA repose sur la difficulté de calculer le logarithme discret d'un grand nombre entier.

6.3 Signature

Les signatures numériques et la cryptographie à clé publique sont au cœur de pratiquement tout ce qui se passe sur une blockchain comme ethereum.

Nous interagissons avec Ethereum à l'aide d'un portefeuille qui est associé à deux clés : une clé publique (adresse de portefeuille) et une clé privée.

En utilisant la cryptographie, il est possible pour une personne de prouver qu'elle détient la clé privée d'une adresse de portefeuille particulier sans révéler la clé elle-même.

La signature cryptographique numérique permet au :

- Signataire de signer un message à l'aide d'une clé privée et diffuser le message signé.
- Il est impossible de récupérer la clé privée en regardant simplement le message et/ou la clé publique.
- Il est possible de vérifier que le signataire détient la bonne clé privée en utilisant la clé publique (adresse du portefeuille).

Les deux figures présentent la signature de la ressource téléchargée en pinata et la signature du NFT après une interaction avec le portefeuille du propriétaire :

```

{
  address: "0x81683f9885593a5c8f6deee7130c89c4fadb1f8a",
  nft: {
    name: "BechirTest",
    description: "test",
    attributes: [
      {trait_type: "attack", value: "0"},
      {trait_type: "health", value: "0"}
    ],
    image: "https://gateway.pinata.cloud/ipfs/QmV7uuRffKBqQPvUy3hZDtpD1vb8M74Y5wTFwVJLN1J2Kk",
    name: "BechirTest",
    signature: "0xa3e283f8be0bb51b10fcbcefc11ae3183e767b843fee052c18c4532a6cecaa07f376f32180e476205e6995a03685d610714d74fc5083be64caadb65e3a2566a1b"
  }
}

```

Figure 5- Signature de la ressource téléchargée en PINATA

```

{
  address: "0x81683f9885593a5c8f6deee7130c89c4fadb1f8a",
  bytes: {0: 255, 1: 216, 2: 255, 3: 224, 4: 0, 5: 16, 6: 74, 7: 70, 8: 73, 9: 70, 10: 0, 11: 1, 12: 1, 13: 0},
  contentType: "image/jpeg",
  fileName: "9",
  signature: "0xb02473ee7ced0b71f02b41037e0dc9592e6f970ed50885de397f1113828d930152271c6cd0159daa66a59df7920fc2918cc1318aef8dad8175bfeefa75986ed81b"
}

```

Figure 6- Signature du NFT

6.4 Vérification des signatures

La vérification des signatures est extrêmement simple grâce à la bibliothèque ⁶ECDSA d'openZeppelin.

L'intérêt d'un NFT est la vérification numérique et le contrôle de ce qui pourrait être un actif physique ou numérique. Si nous ne pouvons pas vérifier l'actif sous-jacent lui-même de la même manière que pour vérifier la propriété du jeton qui représente l'actif, nous avons perdu la trace de l'objectif ultime.

La solution aux deux problèmes est IPFS. IPFS est un réseau de stockage distribué. Cela fonctionne de la même manière que le stockage en nuage. Vous faites une demande de contenu et ce contenu est renvoyé. Cependant, la grande différence est que le contenu est stocké en utilisant un réseau mondial de fournisseurs de stockage. IPFS exploite un outil appelé adressabilité du contenu. Cela signifie qu'au lieu de faire une demande à ce centre de données dans l'Ohio pour un élément de contenu, vous feriez plutôt une demande pour le contenu lui-même. Il pourrait être situé dans l'Ohio. Il peut être situé plus près. Grâce à l'adressabilité du contenu, vous n'avez plus besoin

⁶ ECDSA : Algorithme de signature électronique

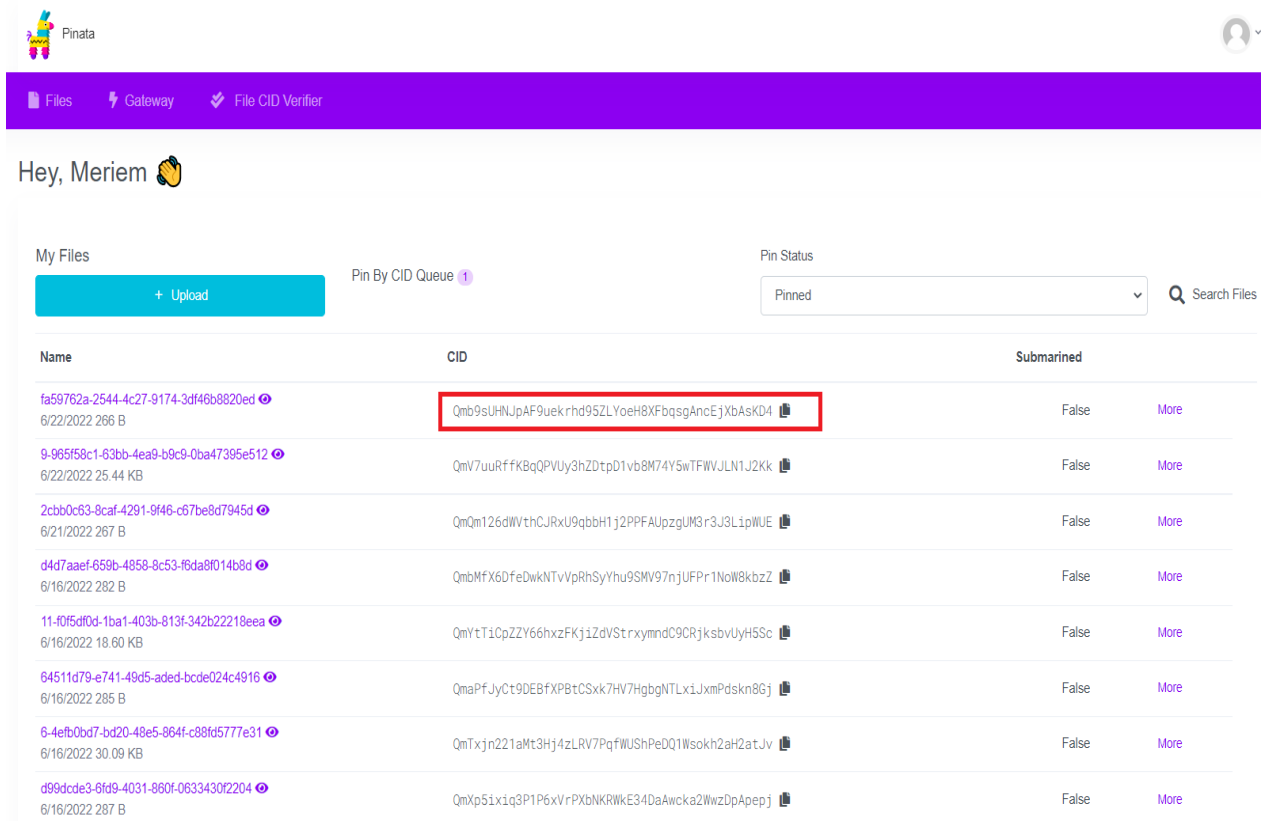
de compter sur un emplacement unique pour la récupération du contenu. Ceci est beaucoup plus efficace pour les blockchains mondiales.

IPFS s'occupe également de la vérifiabilité pour nous. Étant donné que tout le contenu est défini et stocké en fonction du contenu lui-même, si un élément de contenu était falsifié ou modifié, nous aurions une incompatibilité lorsque nous vérifions le contenu et saurons qu'il est erroné. Rendons cela un peu plus clair avec un exemple simple :

Alice stocke une image d'un chat sur IPFS et cette image de chat est représentée par un identifiant de contenu. Pour simplifier, disons que l'identifiant est "C".

Bob demande cette photo de chat et dessine ensuite une moustache sur ce pauvre chat. Lorsque Bob téléchargera sa photo, il n'aura plus le même identifiant. Parce qu'il a changé les données sous-jacentes (de chat à chat moustachu), l'identifiant de Bob pourrait être "M".

Si Bob essayait de faire passer sa photo pour Alice, quiconque prendrait la peine de vérifier saurait qu'il mentait. L'identifiant d'Alice ne correspond pas à celui de Bob et, par conséquent, l'image que Bob essaie de faire passer pour celle d'Alice est manifestement fausse.



Name	CID	Submarined
fa59762a-2544-4c27-9174-3df46b8820ed 6/22/2022 266 B	Qmb9sUHNJpAF9uekrhd95ZLYoeH8XfbqsgAncEjXbAskD4	False More
9-965f58c1-63bb-4ea9-b9c9-0ba47395e512 6/22/2022 25.44 KB	QmV7uuRffKBQPVUy3hZDtpD1vb8M74Y5wTFWVJLN1J2Kk	False More
2cbb0c63-8caf-4291-9f46-c67be8d7945d 6/21/2022 267 B	QmQm126dWVthCJRxu9qbbH1j2PPFAUpzgUM3r3J3LipWUE	False More
d4d7aaef-659b-4858-8c53-f6da8f014b8d 6/16/2022 282 B	QmbMfX6DfeDwkNTvVpRhSyhu9SMV97njUFPr1NoW8kbzZ	False More
11-f0f5df0d-1ba1-403b-813f-342b22218eea 6/16/2022 18.60 KB	QmYtTiCpZZY66hXzFKjizdVStrxymndC9CRjksbvUyHSSc	False More
64511d79-e741-49d5-aded-bcde024c4916 6/16/2022 285 B	QmaPfJyCt9DEBFXPBtCSxk7HV7HgbgNTLxiJxmPdskn8Gj	False More
6-4efb0bd7-bd20-48e5-864f-c88d5777e31 6/16/2022 30.09 KB	QmTxjn221aMt3Hj4zLRV7PqfWUShPeDQ1Wsoh2ah2atJv	False More
d99dcd3-8fd9-4031-860f-0633430f2204 6/16/2022 287 B	QmXp5ixiq3P1P6xVrPXbNKRWKE34DaAwcka2WwZdpApepJ	False More

Figure 7- Ressource stockée en PINATA

7. Choix Technique

Dans cette partie, nous présentons les différents outils utilisés pour la réalisation de l'application. Nous introduisons, tout d'abord les outils utilisés dans la réalisation de la partie frontale de notre application, ensuite, ceux utilisés dans la partie « backend » et enfin les différentes bibliothèques utilisées.

7.1 Technologies frontend

Dans cette section, nous présentons les outils utilisés dans le développement de la partie frontale de notre application : ReactJS, NextJS, TailWind.

7.1.1 React.JS

React est une bibliothèque de développement d'interface utilisateur basée sur JavaScript. Facebook et une communauté de développeurs open source l'exécutent. Bien que React soit une bibliothèque plutôt qu'un langage, il est largement utilisé dans le développement Web. La bibliothèque est apparue pour la première fois en mai 2013 et est maintenant l'une des bibliothèques frontales les plus couramment utilisées pour le développement Web.

- Création facile d'applications dynamiques. React facilite la création d'applications Web dynamiques car il nécessite moins de codage et offre plus de fonctionnalités, contrairement à JavaScript, où le codage devient souvent complexe très rapidement.
- Performances améliorées. React utilise Virtual DOM, créant ainsi des applications Web plus rapidement.
- Composants réutilisables. Les composants sont les éléments constitutifs de toute application React, et une seule application se compose généralement de plusieurs composants. Ces composants ont leur logique et leurs contrôles, et ils peuvent être réutilisés dans toute l'application, ce qui réduit considérablement le temps de développement de l'application.
- Flux de données unidirectionnel
- Il peut être utilisé pour le développement d'applications Web et mobiles
- Outils dédiés pour un débogage facile

Les raisons ci-dessus justifient la popularité de la bibliothèque React et pourquoi nous l'avons adoptée dans le développement de notre application

7.1.2 Next.Js

Next JS, ce framework React orienté Server Side Rendering, s'est imposé parmi les frameworks JavaScript les plus populaires. Il apporte beaucoup de nouvelles fonctionnalités telles que :

- Le déploiement en CLI et l'hébergement gratuit.
- Vitesse de chargement du site rapide.
- Temps de chargement optimisé.

7.1.3 TailWind CSS

Tailwind CSS est un framework permettant aux développeurs de personnaliser totalement et simplement le design de leur application ou de leur site web. Avec ce framework CSS, il est possible de créer un design d'interface au sein même du fichier HTML. Il apporte beaucoup de nouvelles fonctionnalités telles que :

- Un gain de temps grâce aux class génériques.
- Des performances optimales.
- Le support de méthodes avancés.

7.1.4 Web3 JS - API JavaScript Ethereum

Web3.js est une collection de bibliothèques qui vous permettent d'interagir avec un nœud ethereum local ou distant en utilisant HTTP, IPC ou WebSocket.

7.2 Comparaison entre ReactJs et Angular

Angular est un framework javascript complet développé par google, publiée en 2010. Il permet aux développeurs d'utiliser HTML comme langage de modèle et permet à la syntaxe HTML d'exprimer brièvement et clairement les composants de l'application.

Le tableau 4 explique la différence entre ReactJS et Angular.

Propriété	ReactJS	Angular
Type	React est une bibliothèque JavaScript, et elle est beaucoup plus ancienne que Angular.	Angular est un framework complet.
Utilisation des bibliothèques	React js peut être empaqueté avec d'autres bibliothèques de programmation.	Angular est une solution complète en soi.
Courbe d'apprentissage	Il est plus facile à saisir par rapport à Angular. Cependant, il est difficile d'apprendre lorsqu'il est augmenté avec Redux.	Apprendre Angular n'est pas facile pour les débutants. Ainsi, il nécessite beaucoup de formation.

Temps d'installation	React prend plus de temps à se mettre en place. Mais, il est très rapide pour livrer des projets et créer des applications.	Angular est facile à configurer mais peut entraîner une augmentation du temps de codage, ce qui entraîne également des retards dans la livraison des projets.
Écrit en	Javascript	TypeScript
Types d'applications	Utilisez cette application si vous souhaitez développer des applications natives, des applications hybrides ou des applications Web	Vous devez utiliser ce cadre si vous souhaitez développer une SPA (application à page unique) et des applications mobiles.
Ajout de la bibliothèque Javascript au code source	Possible	Pas possible
Modèle	Il est basé sur Virtual DOM	Basé sur MVC (Model View Controller)
Abstraction	Fort	Moyen

Tableau 4- Tableau comparatif entre ReactJs et Angular

7.3 Technologies backend

Dans cette section, nous présentons les outils de développement de la partie « backend » de notre application qui sont : Solidity, Ganache, Truffle.

7.3.1 Solidity

Solidity est un langage de programmation orienté objet, haut niveau, compilé de bas niveau (bytecodes) interprété par l'environnement d'exécution d'Ethereum pour écrire des contrats intelligents.

7.3.2 Truffle

Truffle est l'un des frameworks les plus populaires pour écrire, déployer et tester des contrats intelligents. Il prend en charge les blockchains EVM telles que Ethereum, Polygon et Binance Smart Chain.

7.3.3 Ganache

Ganache est un simulateur de blockchain qui peut être mis en place localement. Les dapps peuvent être développées, testées et déployées via Ganache.

8. Environnement de travail

Dans cette partie, nous présentons l'environnement matériel et logiciel utilisé dans la réalisation de notre application.

8.1 Environnement matériel

Pour la réalisation de notre application, nous utilisons un ordinateur portable dont les caractéristiques sont résumées dans le tableau 3

Machine	Caractéristiques
Lenovo IdeaPad L 340 GAMING	Système d'exploitation : Windows 10 professionnel 64 bits Ecran : 15.6" Full HD IPS Processeur : Intel Core i5-9300H Mémoire : 16 Go Disque dur : SSD 1TO Carte graphique : Nvidia GeForce GTX 1650

Tableau 5 - Environnement matériel

8.2 Environnement logiciel

Dans cette partie, nous présentons les différents logiciels utilisés pour la réalisation de notre application.

8.2.1 Visual studio code

Visual Studio Code est un éditeur de code source développé par Microsoft pour Windows, Linux et macOS. Il prend en charge le débogage, le contrôle Git (un logiciel libre de gestion de versions décentralisé) intégré et GitHub, la mise en évidence de la syntaxe, la complétion intelligente du code, les fragments et le remaniement du code. Visual Studio Code se caractérise par l'extensibilité. Il donne la possibilité d'ajouter d'autres extensions qui facilitent la tâche de développement.

8.2.2 Windows PowerShell

Windows PowerShell est une interface en ligne de commande. Nous utilisons Windows PowerShell pour exécuter les CLIs (Command Line Interface) de Truffle.

8.2.3 Postman

Postman est une extension pour le navigateur Google Chrome qui permet d'envoyer des requêtes et qui nous permet de tester les APIs que nous implémentons.

8.2.4 GitBash

Git Bash est une application pour les environnements Microsoft Windows OS qui fournit des utilitaires et une expérience shell basée sur Unix pour les commandes de ligne de commande Git. Un shell est une application informatique qui s'intègre au système d'exploitation et expose ses services à un utilisateur final ou à d'autres applications.

9. Architecture physique

La figure 4 présente l'architecture physique de notre application. L'architecture adoptée divise notre application en deux couches :

- La première couche est la couche client, elle est composée de : l'application web ainsi que de l'interface de notre application d'authentification par blockchain.
- La deuxième couche qui est la couche traitement. Elle présente notre contrat intelligent développé en solidity qui est le cœur de notre application et qui communique via des bytécodes avec la machine virtuelle d'ethereum.

Ces deux couches communiquent entre elles grâce au protocole RPC (remote procedure call)

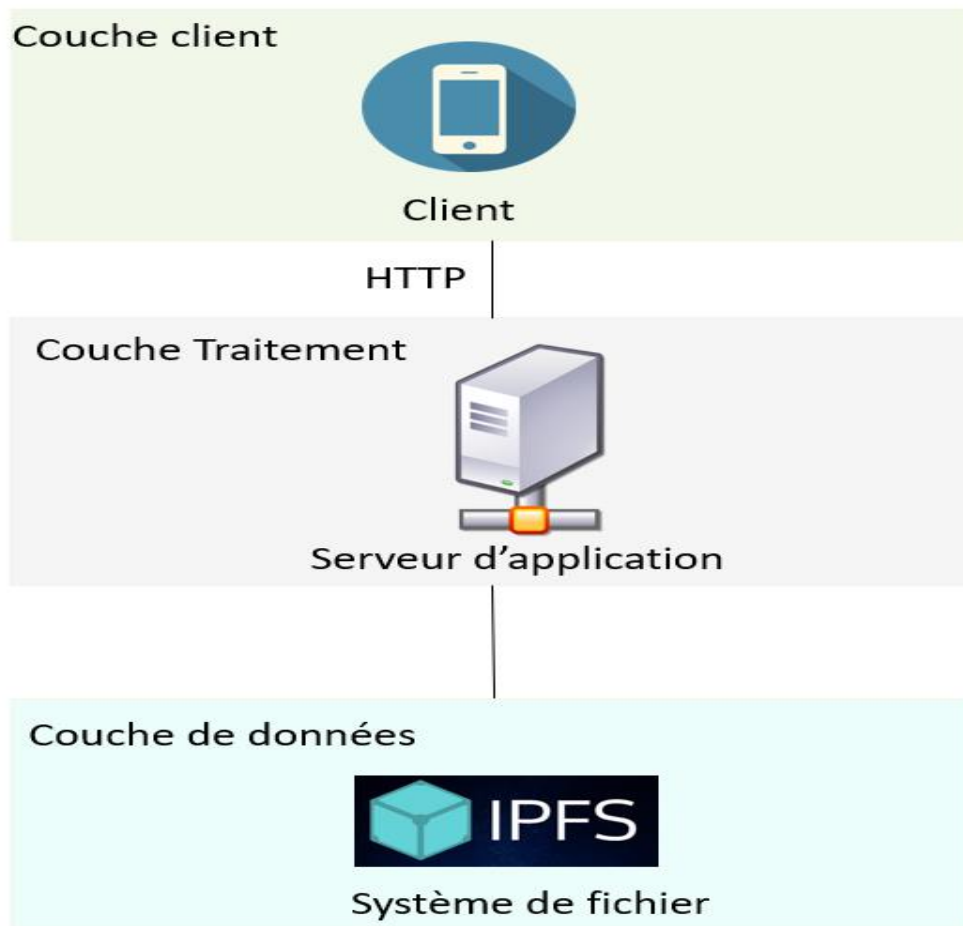


Figure 8 - Architecture physique Otentix

Conclusion

Dans ce chapitre nous avons cité les différentes étapes pour accomplir le premier sprint qui est la mise en place des fonctionnalités basiques dans le contrat intelligent avec la connexion à la blockchain, les étapes qu'on a respecté afin de livrer la première architecture du projet, ainsi une explication des outils utilisés lors du développement.

Le chapitre qui suit consacrera au développement des nouvelles fonctionnalités avancées.

Chapitre 4 : Sprint 2 : Développement des fonctionnalités de l'application

1.Introduction

Dans le chapitre précédent nous avons présenté le premier sprint, qui est la base de notre projet. Nous passons à la partie développement et intégration des fonctionnalités qui va contenir toutes les tâches dans le product backlog dans le tableau ci-dessous.

ID	User Stories	Priorité
1	En tant qu'utilisateur, Je peux me connecter et consulter mon profil	Moyenne
2	En tant qu'utilisateur, Je peux consulter la liste de mes NFT.	Moyenne
3	En tant qu'utilisateur, Je peux acheter un NFT.	Elevée
4	En tant qu'administrateur, Je peux créer des NFT.	Elevée
5	En tant qu'administrateur, Je peux signer des NFT.	Elevée
6	En tant qu'administrateur, Je peux publier un NFT (le placer dans la marketplace)	Elevée
7	En tant qu'administrateur, Je peux générer un QrCode à des NFT.	Moyenne
8	En tant qu'administrateur, Je peux bruler ou modifier un NFT.	Elevée
9	En tant qu'administrateur, Je peux préciser un prix pour chaque NFT.	Elevée
10	En tant qu'administrateur, Je peux modifier le prix assigner à un NFT.	Faible

Tableau 6 - Product backlog du sprint 2 : Développement des fonctionnalités de l'application OTENTIX

2.Architectures logiques

La figure ci-dessous présente l'architecture de notre application. Il est important d'avoir une idée du fonctionnement des applications dans le monde web3. Les couches d'une application décentralisée sont un peu plus complexes. Mais pour simplifier, voici comment modéliser l'architecture d'une dApp :

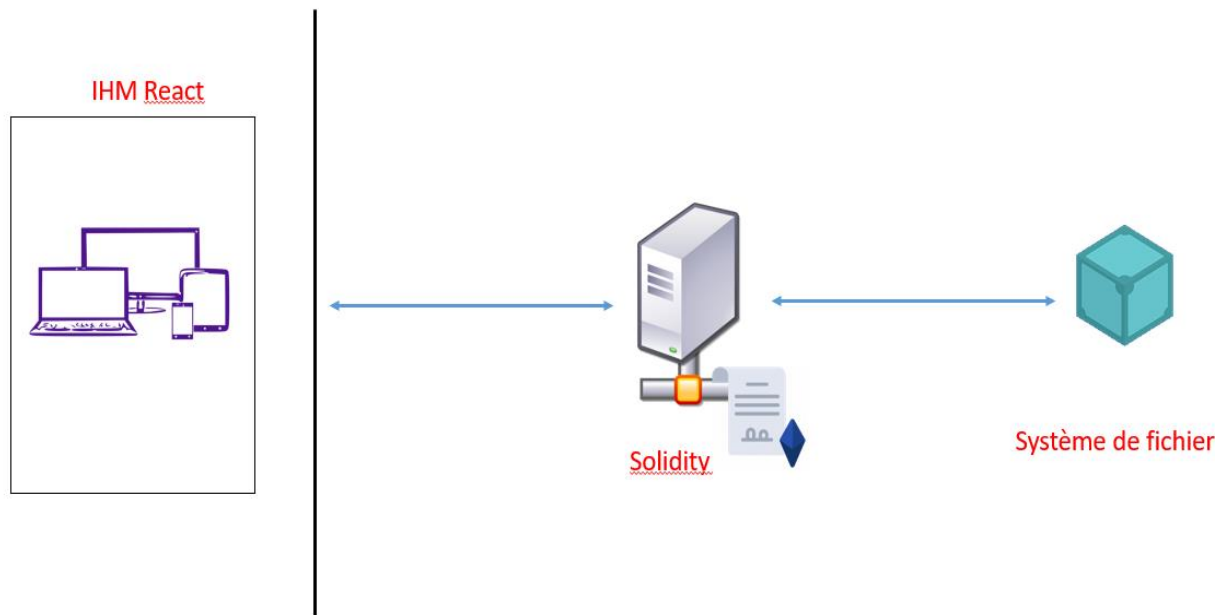


Figure 9 - Architecture logique de l'application

Notre application web interagit directement avec la blockchain et gère tous les appels au contrat intelligent via web3js dans le but de rendre le contrat intelligent lisible par l'utilisateur en créant le front-end et en le connectant à un réseau de test Ethereum à l'aide d'une bibliothèque wallet et web3.

3. Développement du module NFT

C'est l'une des parties les plus importantes dans le projet, d'où l'administrateur consulte cette page pour avoir la liste des NFT, là où il peut transférer un NFT à un autre compte, brûler un NFT, générer un QrCode à un NFT, ou aussi ajouter un NFT à IPFS.

3.1 Scénario du cas d'utilisation consulter NFT

Tous les acteurs de notre application ont accès à ce module après une authentification réussie via son portefeuille.

Quand l'acteur accède à l'interface de consultation des NFT, une demande est envoyée du contrat intelligent au système de fichier interplanétaire (IPFS) pour récupérer cette liste et la renvoyer, par la suite, pour l'affichage au client.

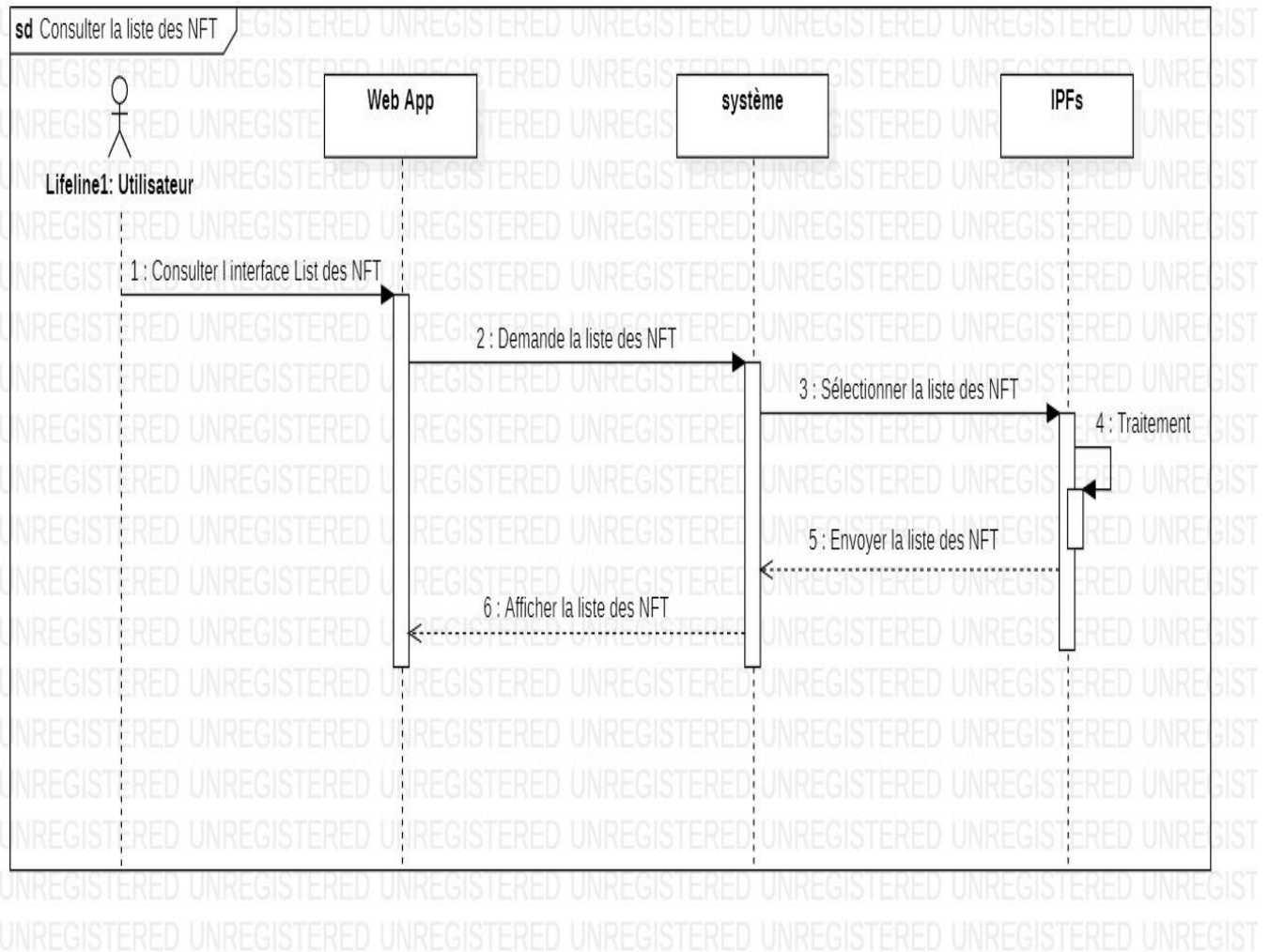


Figure 10 - Diagramme de séquence « Consulter NFT »

3.2 Diagramme de séquence « Ajouter un NFT »

La figure 7 détaille le diagramme de séquence « ajouter un NFT ». Tout d'abord, il faut consulter l'interface d'ajout d'un NFT, ensuite il faut remplir les différents champs affichés dans cette interface.

Nous allons utiliser Pinata pour ajouter notre ressource à IPFS et nous assurer qu'elle reste épinglée. Nous ajouterons également nos métadonnées ⁷JSON à IPFS afin de pouvoir les transmettre à notre contrat de jeton. Une fois les champs renseignés, une demande de validation est renvoyée à l'administrateur. Dans le cas où l'administrateur ne valide pas l'ajout du NFT, aucune modification dans le système de fichier interplanétaire (IPFS) n'est effectuée. Dans le cas de validation de l'ajout, le nouveau NFT sera ajoutée à IPFS.

⁷ JSON : JavaScript Object Notation

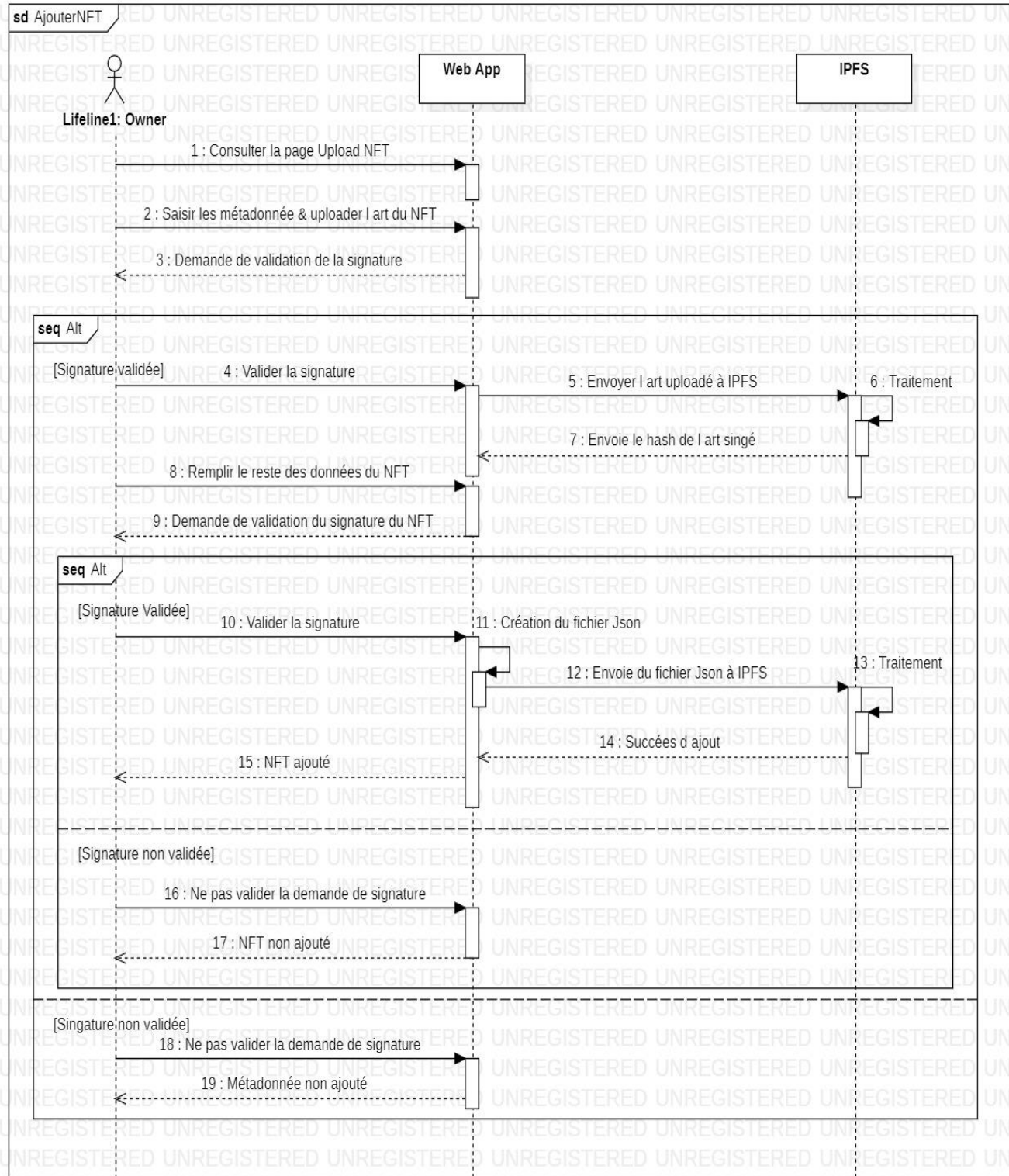


Figure 11 - Diagramme de séquence « ajouter un NFT ».

Pour comprendre pourquoi nous faisons ce que nous faisons, nous devons comprendre comment fonctionne le mint NFT.

Minter un NFT cela signifie que vous écrivez du code (appelé contrat intelligent) qui indique à la blockchain d'initialiser une table pour vous. Cette table stocke les informations de propriété et de métadonnées sur vos NFT. Dans la plupart des cas, au lieu de stocker des données d'un NFT, nous stockons plutôt simplement des données sur le NFT. Ces données (ou métadonnées) sont stockées dans un format appelé JSON. Ce fichier JSON doit contenir des informations sur le NFT telles que son nom, sa description, ⁸l'URL de l'image, ses attributs, etc.

Le stockage des métadonnées dans ce format sur la blockchain est encore très coûteux. Par conséquent, nous ajoutons une couche d'abstraction supplémentaire et téléchargeons également ce JSON dans le cloud et stockons simplement une URL pointant vers le fichier JSON.

```
{
  "name": "BechirTest",
  "description": "test",
  "image": "https://gateway.pinata.cloud/ipfs/QmV7uuRffKBqQPvUy3hZDtpD1vb8M74Y5wTFwVJLN1J2Kk",
  "attributes": [{ "trait_type": "attack", "value": "0" },
    { "trait_type": "health", "value": "0" },
    { "trait_type": "speed", "value": "0" } ]
}
```

Figure 12 - Structure du fichier JSON du NFT

3.3 Architecture Mint NFT

Selon la blockchain, NFT est comme une transaction de métadonnées. Donc, minter un NFT est le processus d'exécution de cette transaction sur la blockchain. La figure ci-dessous présente le mécanisme du mint d'un NFT :

⁸ URL : Uniform Resource Locator.

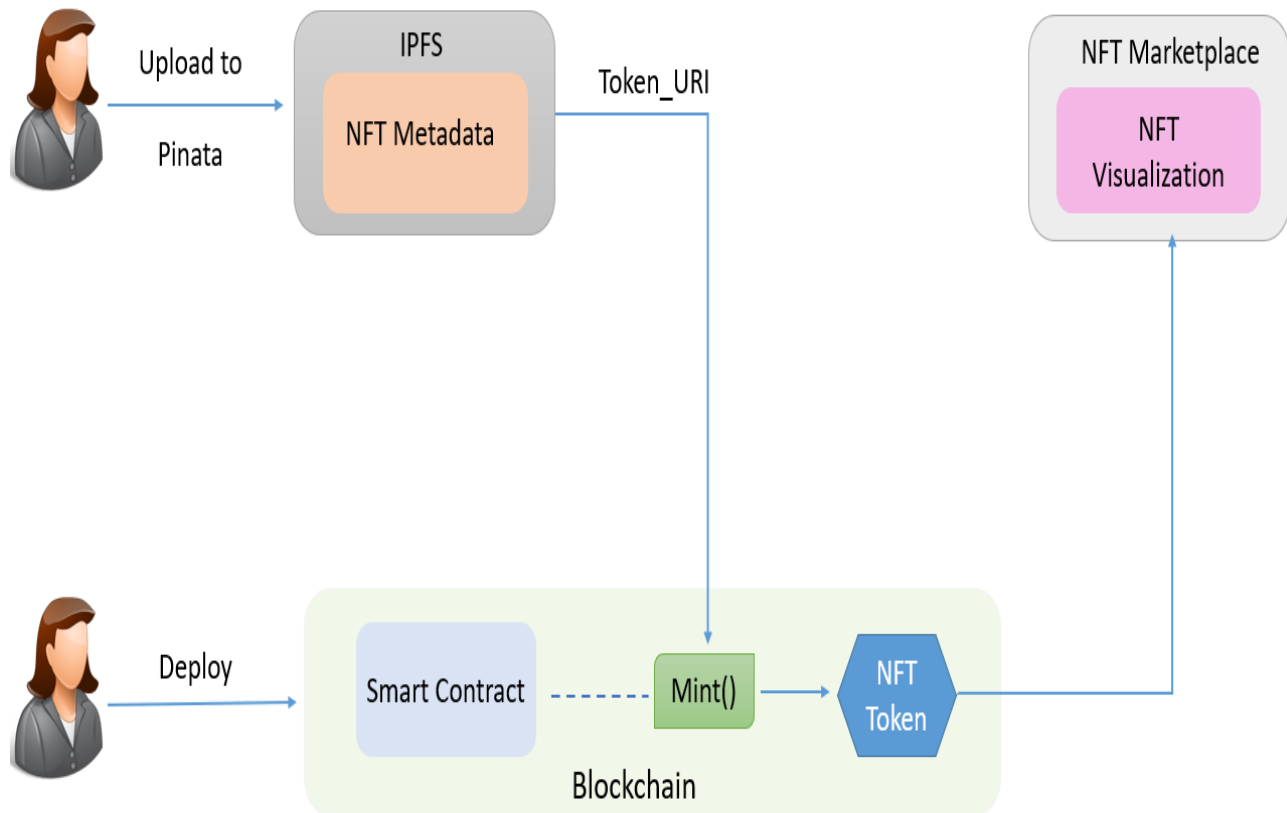


Figure 13 - Architecture Mint NFT

3.4 Diagramme de séquence « Bruler un NFT »

La figure 14 présente la brulure d'un NFT », cette fonctionnalité est accessible seulement pour un administrateur.

Ce dernier commence par consulter l'interface des NFT pour afficher la liste des NFT, puis il va choisir un NFT à bruler. Ensuite, il reçoit un message de demande de validation de suppression. Dans le cas de la confirmation de la suppression, le NFT sélectionné est supprimé de la base de données et ne va plus figurer, par la suite, dans la nouvelle liste des NFT affichés. Dans le cas où l'administrateur ne valide pas la brulure du NFT, aucune modification dans le système de fichier inter-planétaire (IPFS) n'est effectuée.

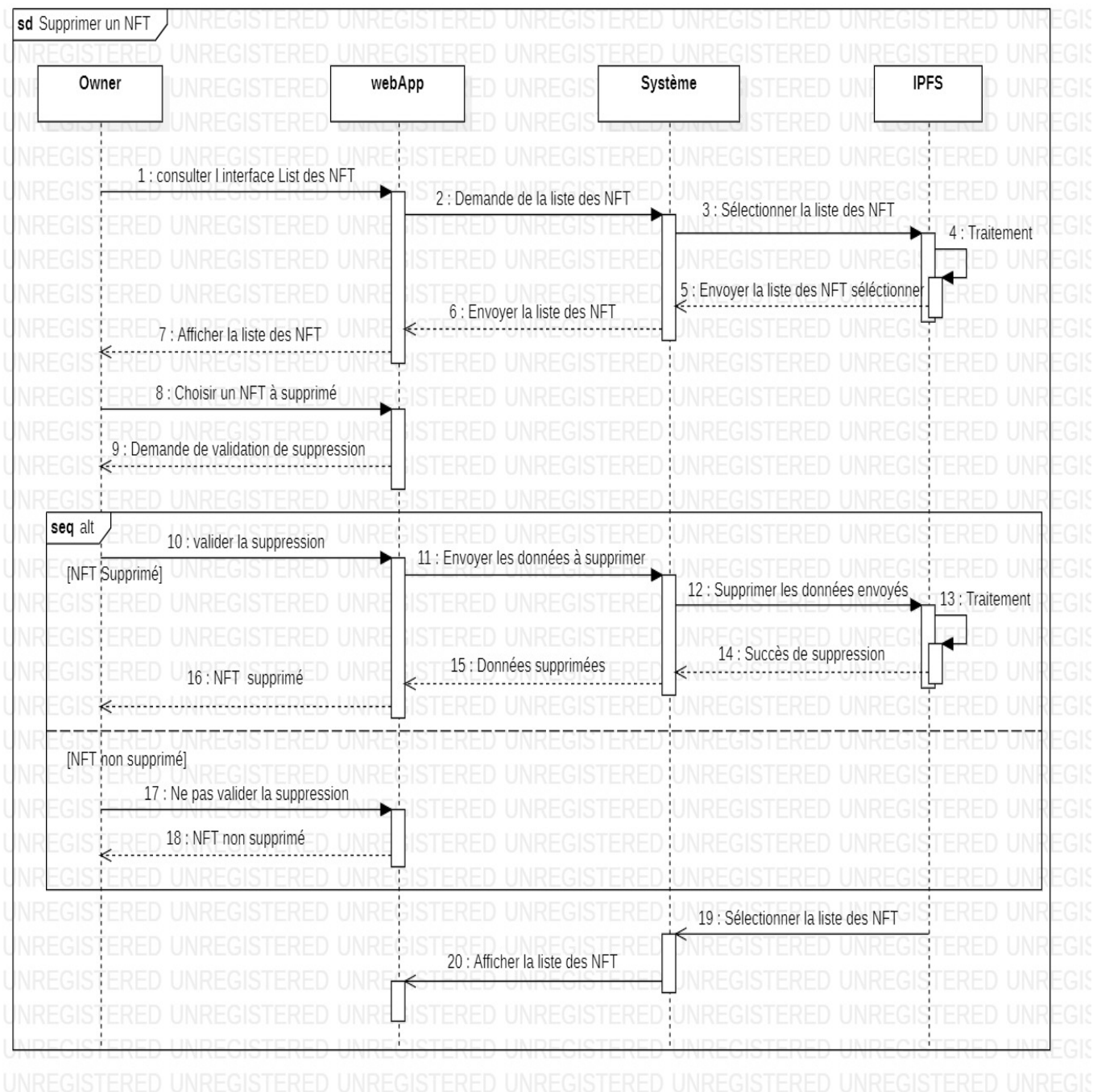


Figure 14 - Diagramme de séquence « bruler un NFT »

3.5 Diagramme de séquence « Transférer un NFT »

La figure 15 présente le transfert d'un NFT, cette fonctionnalité est accessible seulement pour un propriétaire. Ce dernier commence par consulter l'interface des NFT pour afficher la liste des NFT, puis il va choisir un NFT à transférer.

Ensuite, il reçoit un message de demande de validation de transfère. Dans le cas de la confirmation du transfère, le NFT sélectionné est transféré à un autre utilisateur.

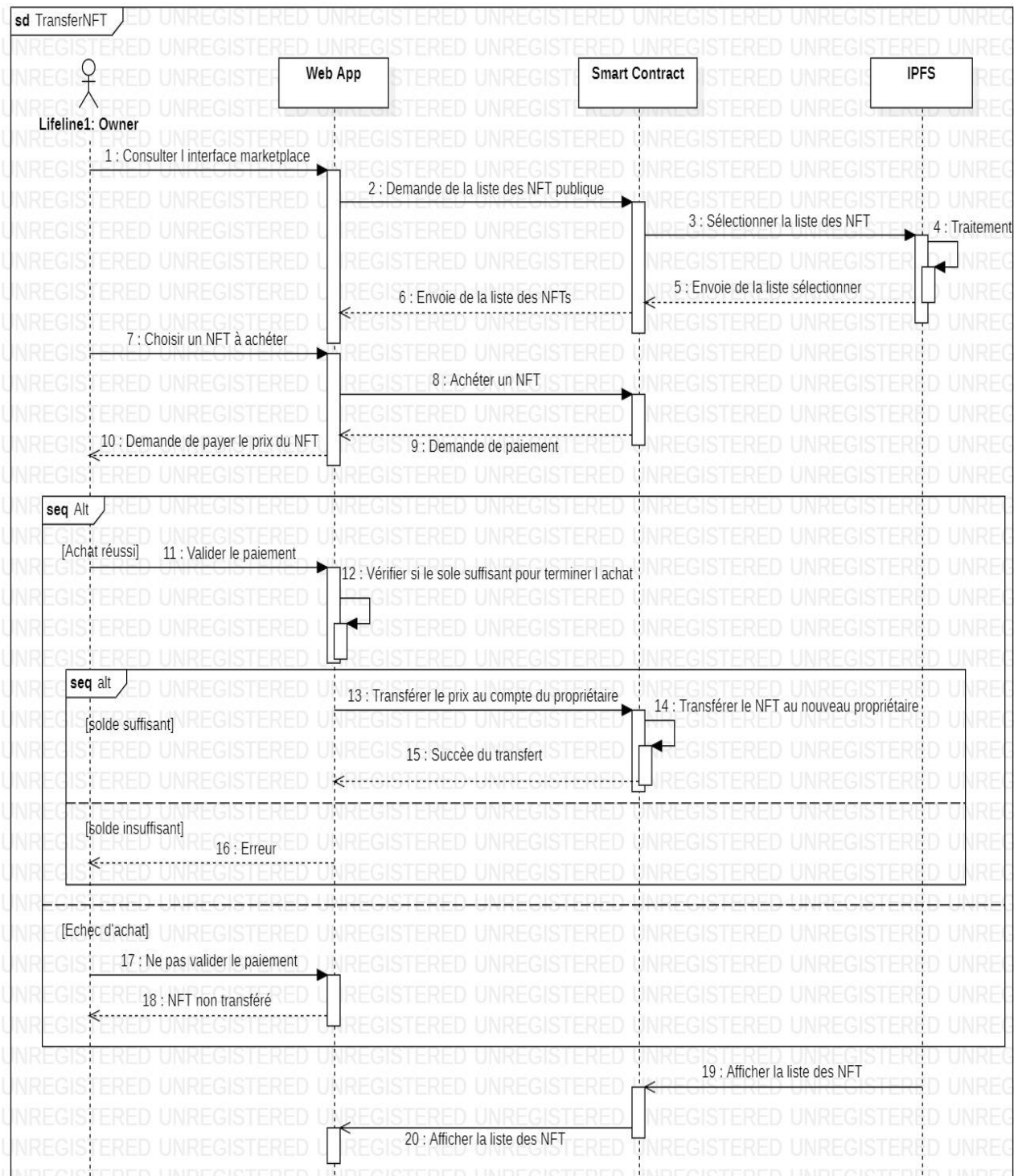


Figure 15 - Diagramme de séquence « transférer un NFT »

3.6 Diagramme de séquence « Générer un QrCode »

La figure 16 présente la génération d'un QrCode, cette fonctionnalité est accessible seulement par le propriétaire du NFT. Ce dernier commence par consulter l'interface des NFT pour afficher la liste des NFT, puis il va choisir un NFT, clique sur le bouton générer un QrCode.

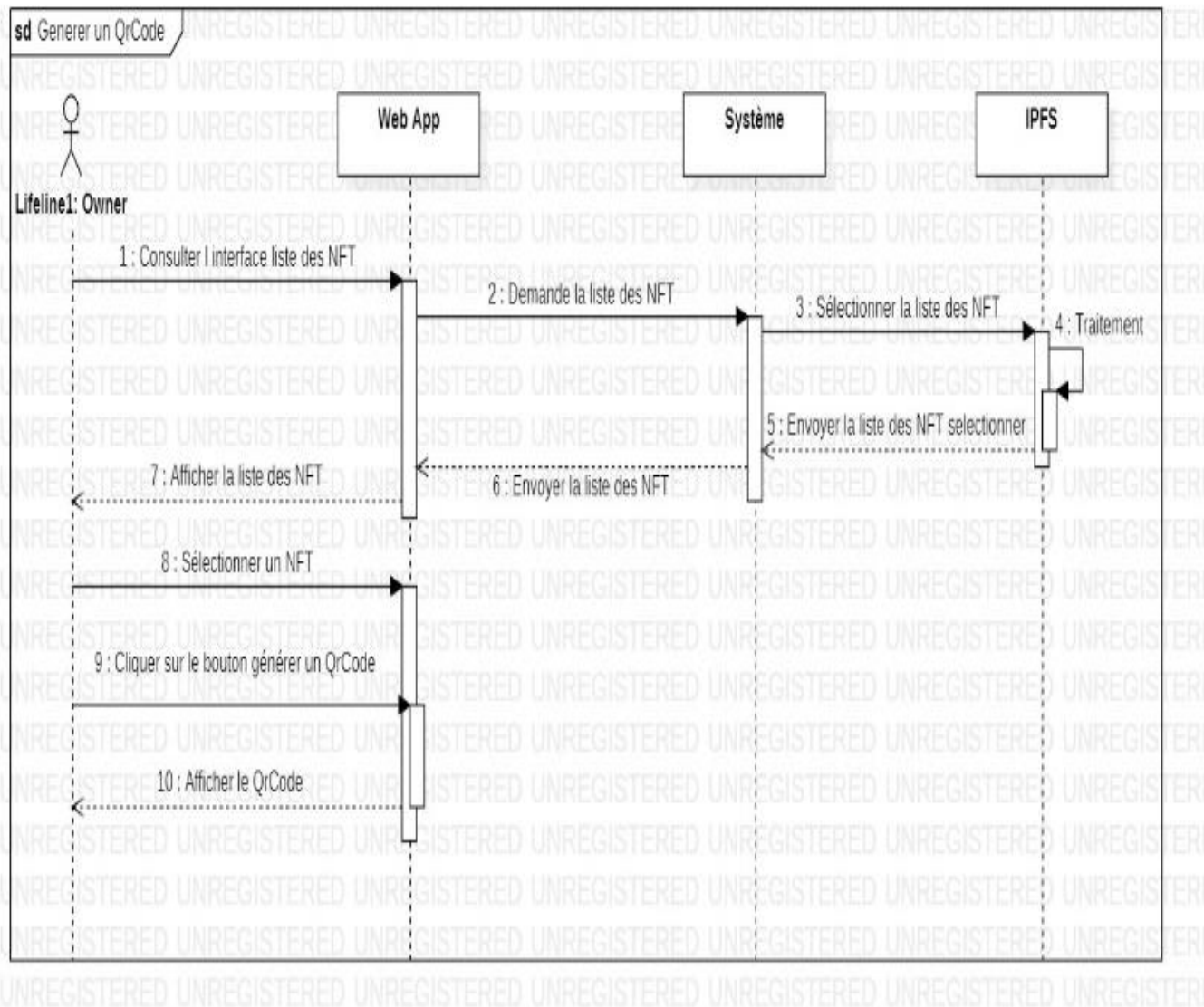


Figure 16- Diagramme de séquence « Générer un Qrcode »

4.Présentation des interfaces graphiques

Dans cette partie, nous présentons les principales interfaces graphiques de notre application.

4.1 Interface d'accueil

Lors de l'ouverture de notre application, la première page que nous rencontrons est la page d'accueil présenté dans la figure 17. L'utilisateur doit d'abord se connecter via son wallet (portefeuille) saisir son mot de passe et en appuyant sur le bouton « déverrouiller », il sera redirigé vers l'interface marketplace.

L'utilisateur utilise l'extension Web qui interagit avec Metamask pour déployer le contrat dans la blockchain Ethereum.

L'extension prévoit le contrat que va être déployé sur le réseau et Metamask s'assure que la transaction est signée et que les fonds nécessaires existent. Il propage ensuite correctement la transaction au réseau Ethereum.

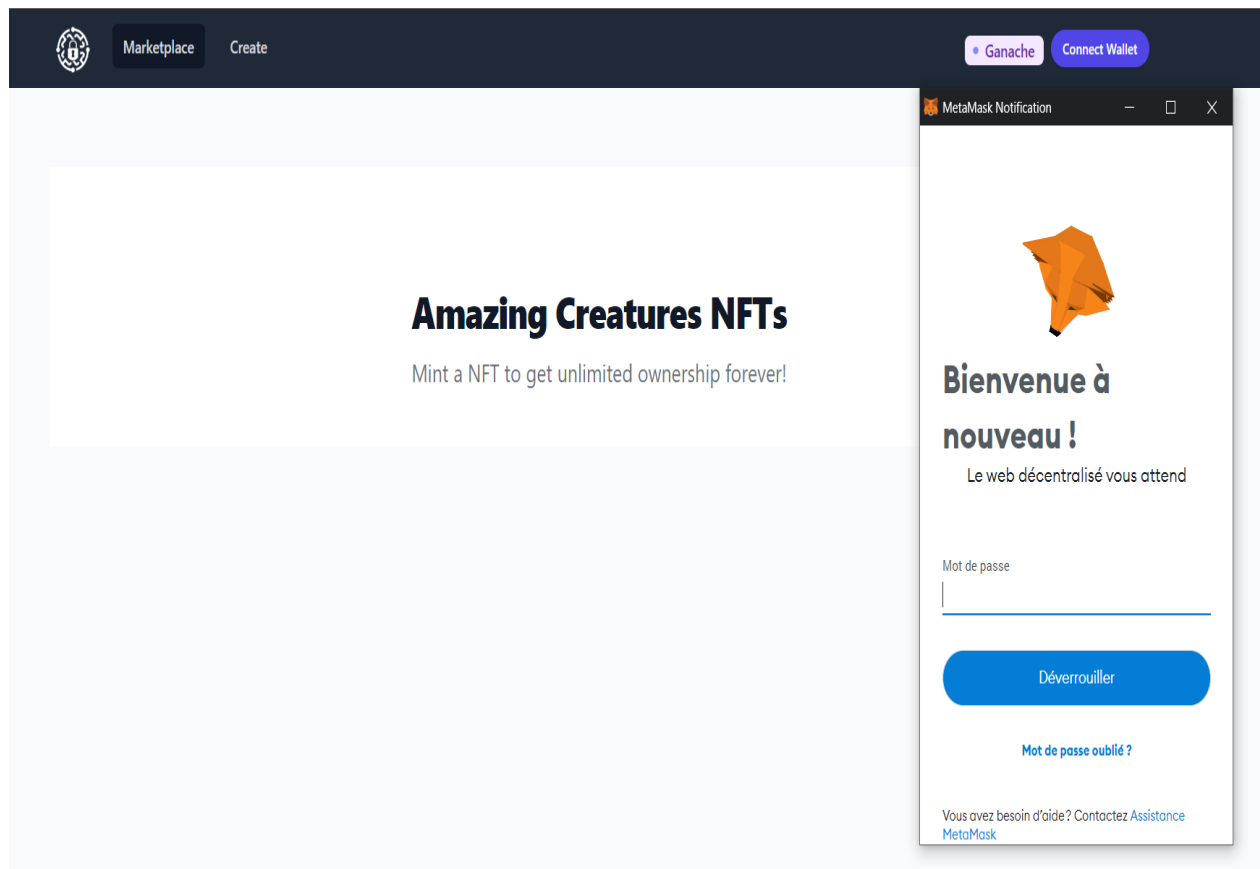


Figure 17 - Interface d'accueil

4.2 Interface Marketplace

Après la phase d'authentification via le portefeuille, la page de liste des NFT est la première page que nous rencontrons. Les différents NFT listés sont accessibles à partir de cette page.

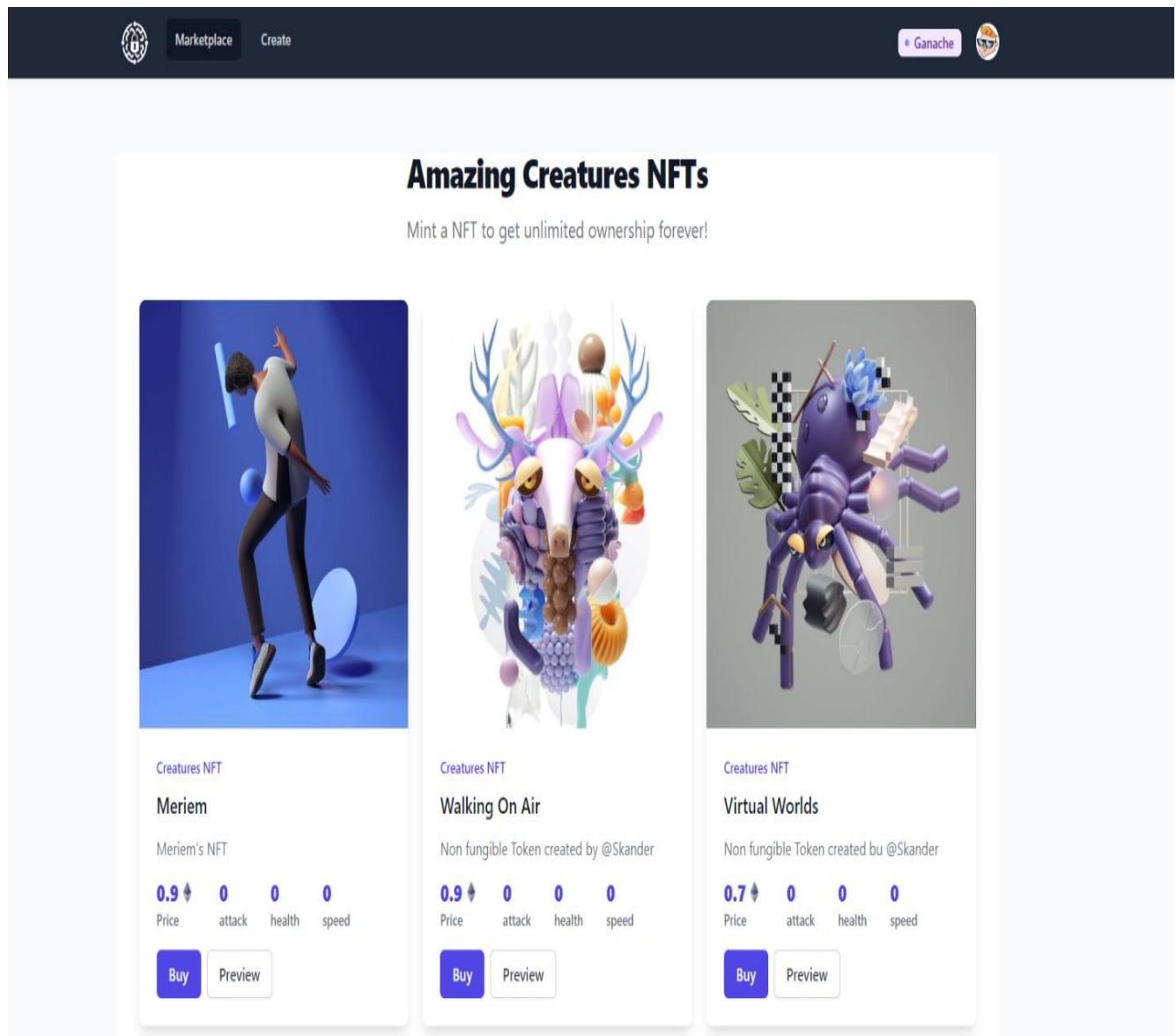


Figure 18 - Interface liste des NFT

4.3 Interface « Buy NFT »

La page marketplace offre la possibilité à l'utilisateur d'acheter un NFT en cliquant sur le bouton « buy NFT », une fenêtre du portefeuille (Metamask) s'ouvre pour accomplir la transaction l'utilisateur doit payer le prix du NFT à acheter de son portefeuille pour qu'il devienne le propriétaire du NFT.

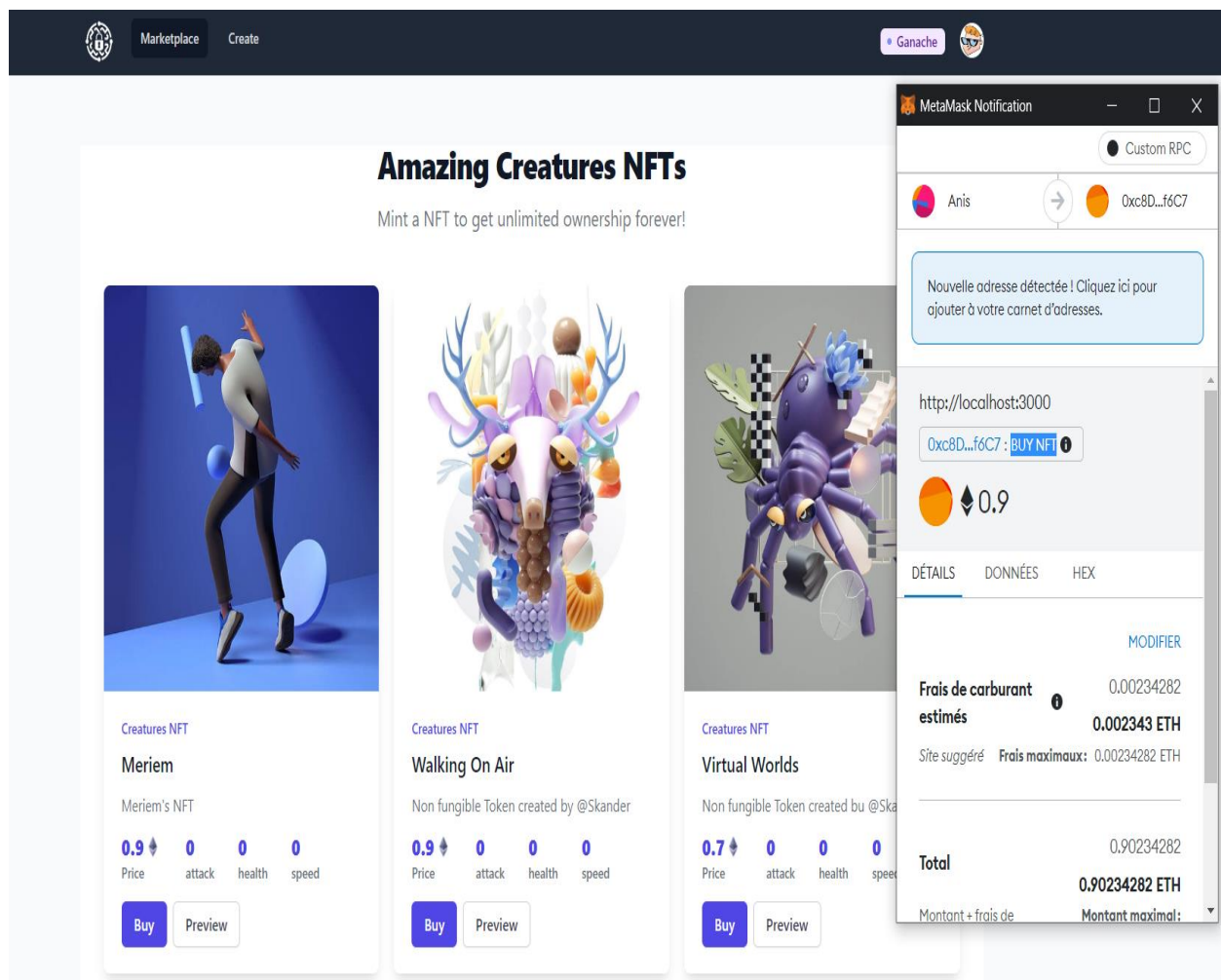


Figure 19- Interface d'achat d'un NFT

Après la validation de cette transaction le NFT en question disparaît de la liste et un message s'affiche dans la table de bord que la transaction a été validée et que cet utilisateur est devenu le propriétaire de cet NFT comme le montre la figure 20.

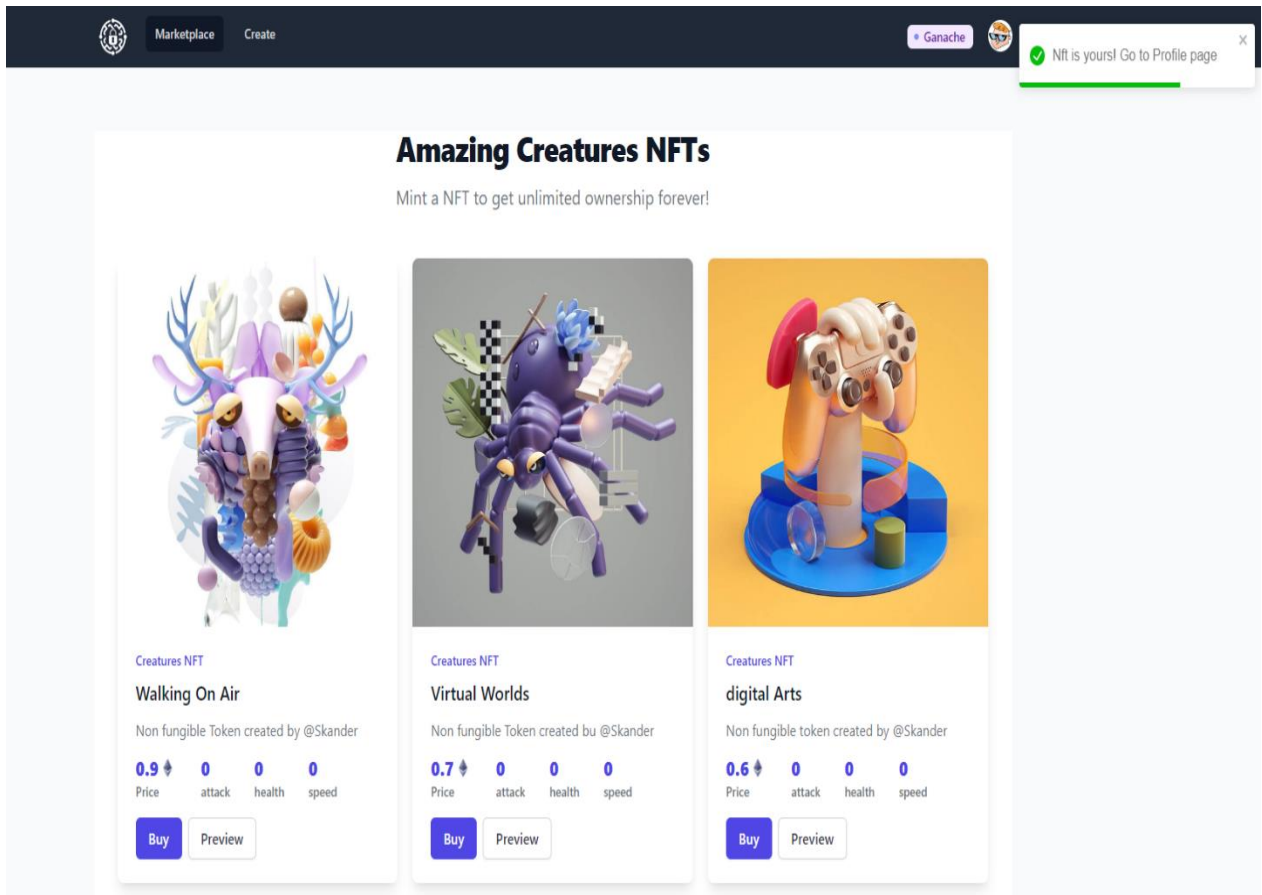


Figure 20- Interface de confirmation de la transaction d'achat d'un NFT

4.4 Interface « Profile »

Lorsque l'utilisateur consulte sa collection des NFT il va trouver le NFT en question ajouté à sa liste. Il va pouvoir consulter chaque NFT et voir ces propriétés, générer un QrCode en cliquant sur le bouton « générer QrCode » ou bien le rendre publique et le placer dans la marketplace en cliquant sur le bouton « list ». Et, si le NFT est déjà listé dans la marketplace le bouton « list » va être désactivé.

Lorsqu'un utilisateur met un NFT en vente, la propriété de l'article sera transférée du créateur au contrat du marché.

Le propriétaire du marché pourra fixer des frais d'inscription. Ces frais seront prélevés sur le vendeur et transférés au propriétaire du contrat à la fin de toute vente, permettant au propriétaire de la place de marché de générer des revenus récurrents à partir de toute vente effectuée.

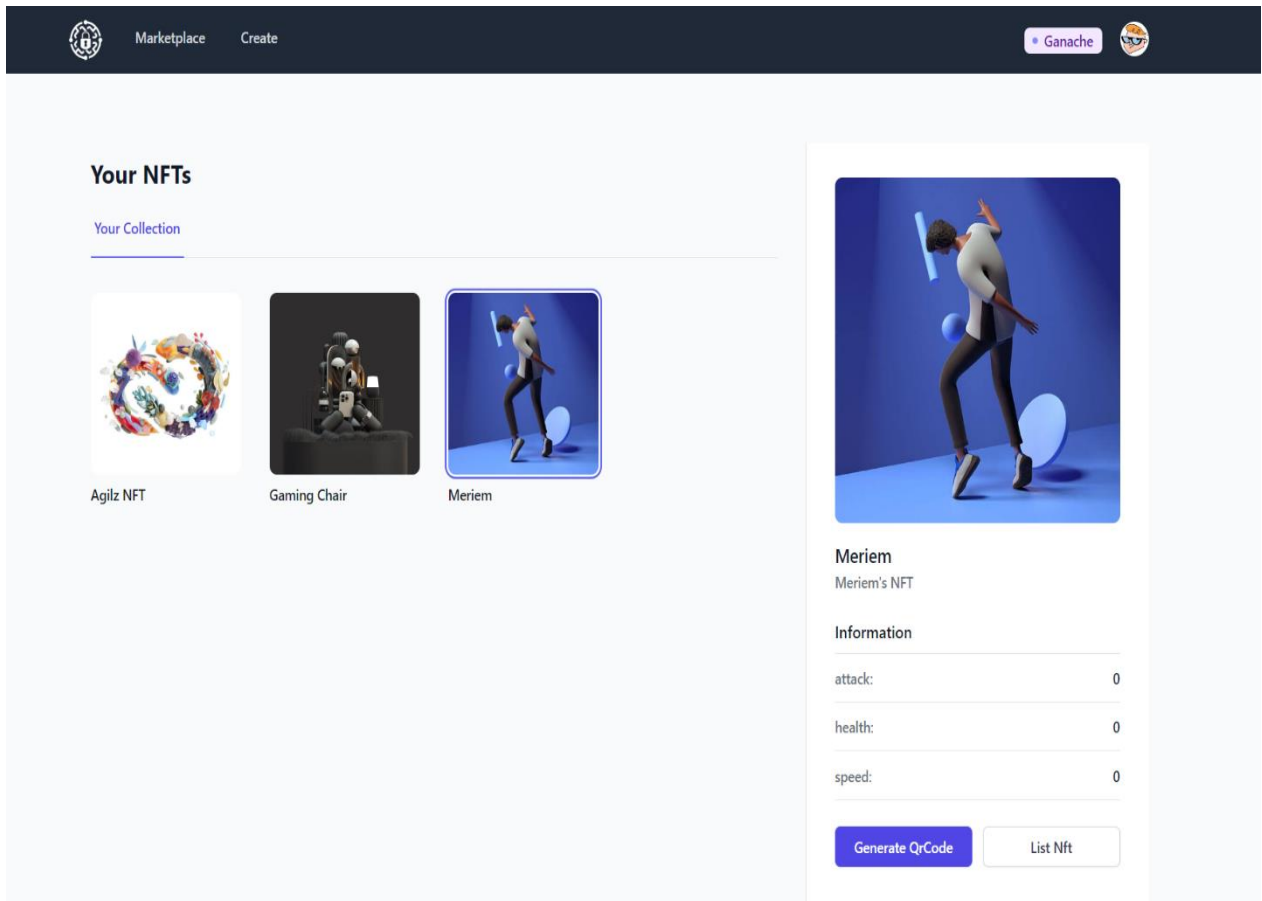


Figure 21- Interface profil utilisateur

4.5 Interface « Créer un NFT »

La figure 22 présente l'interface de création d'un NFT où l'utilisateur peut saisir le nom et un descriptif du NFT et choisir une ressource à uploader dans IPFS, dès qu'il télécharge la ressource une demande de signature de cette image en question est déclenché à travers le portefeuille (Metamask)

Dans cette page l'utilisateur peut :

- Télécharger et enregistrer des fichiers sur PINATA.
- Créer un nouveau NFT.
- Définir les métadonnées et le prix de l'article et le mettre en vente sur le marché.
- Signer un NFT
- Signer la ressource téléchargée sur PINATA

Maintenant notre ressource est enregistrée et prête à être téléchargé, une fois la signature réussie un hachage IPFS (souvent appelé identifiant de contenu ou CID) sera fourni.

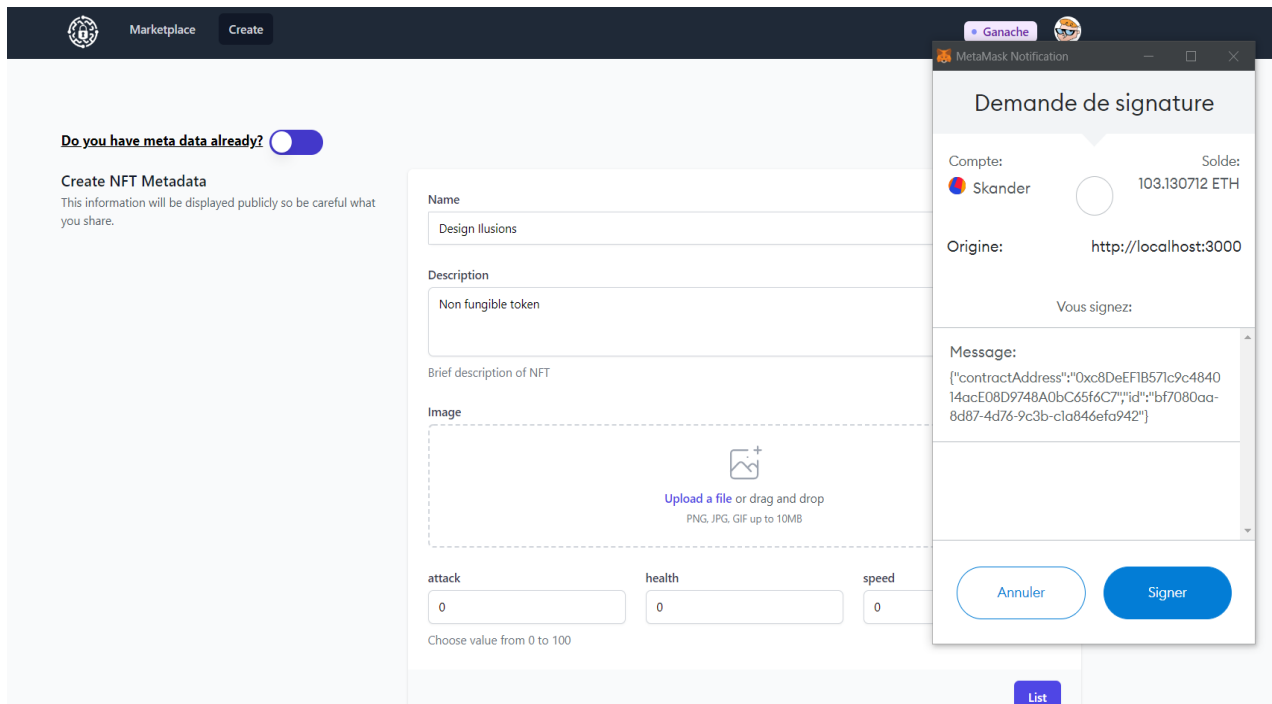


Figure 22- Signature de la ressource téléchargée

Ce hachage est la représentation vérifiable de notre actif. Une fois quelqu'un falsifiait notre actif et le modifiait, le hachage serait différent. Ce hachage va être utilisé lors de l'étape suivante, le mint du NFT.

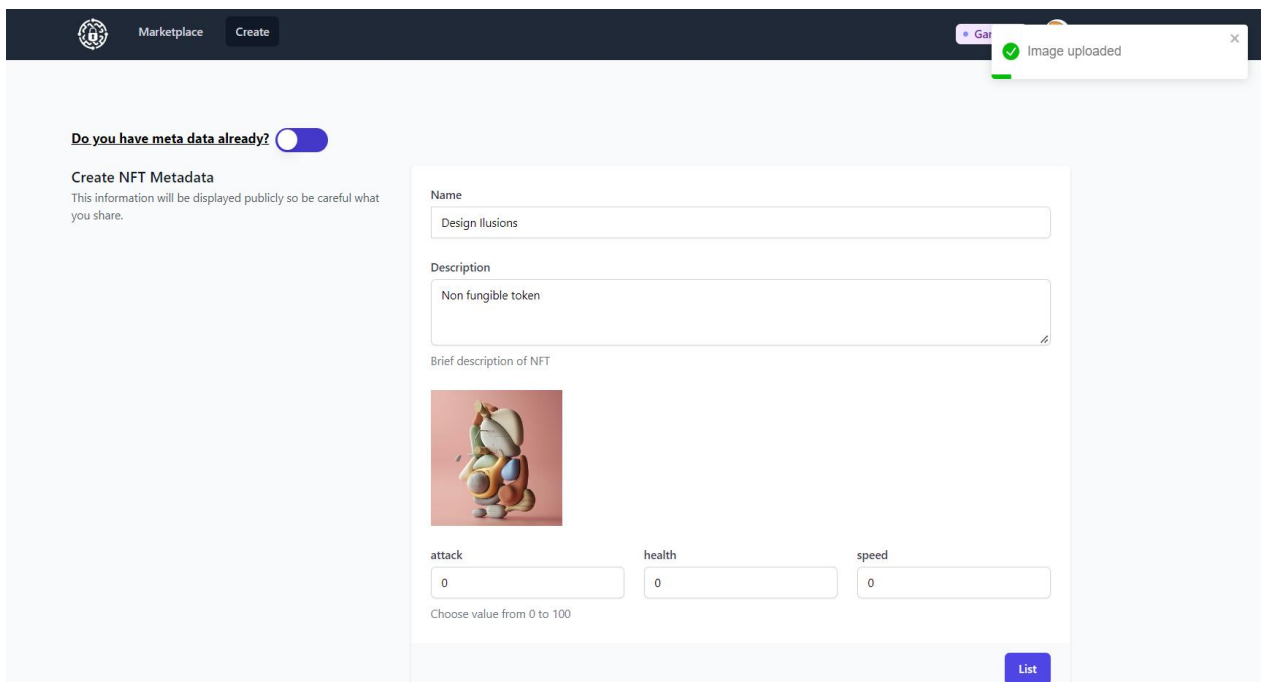


Figure 23- Succès de téléchargement de la ressource signée

Cette fonctionnalité est déclenchée via notre contrat intelligent. Tout hôte IPFS offre une passerelle publique qui pourra afficher le contenu, dans notre cas c'est PINATA.

Une fois l'utilisateur valide et remplit tous les champs nécessaires et après la signature de la ressource quand il clique sur le bouton « list », le portefeuille (Metamask) sera déclenché et demande la validation de la signature du NFT minté. Comme le montre la figure ci-dessous.

L'utilisateur peut aussi lister un NFT existant via l'interface ci-dessous, il doit juste saisir l'URL du NFT stocké dans le pinata, préciser le prix en ethereum et valider en cliquant sur le bouton « list »

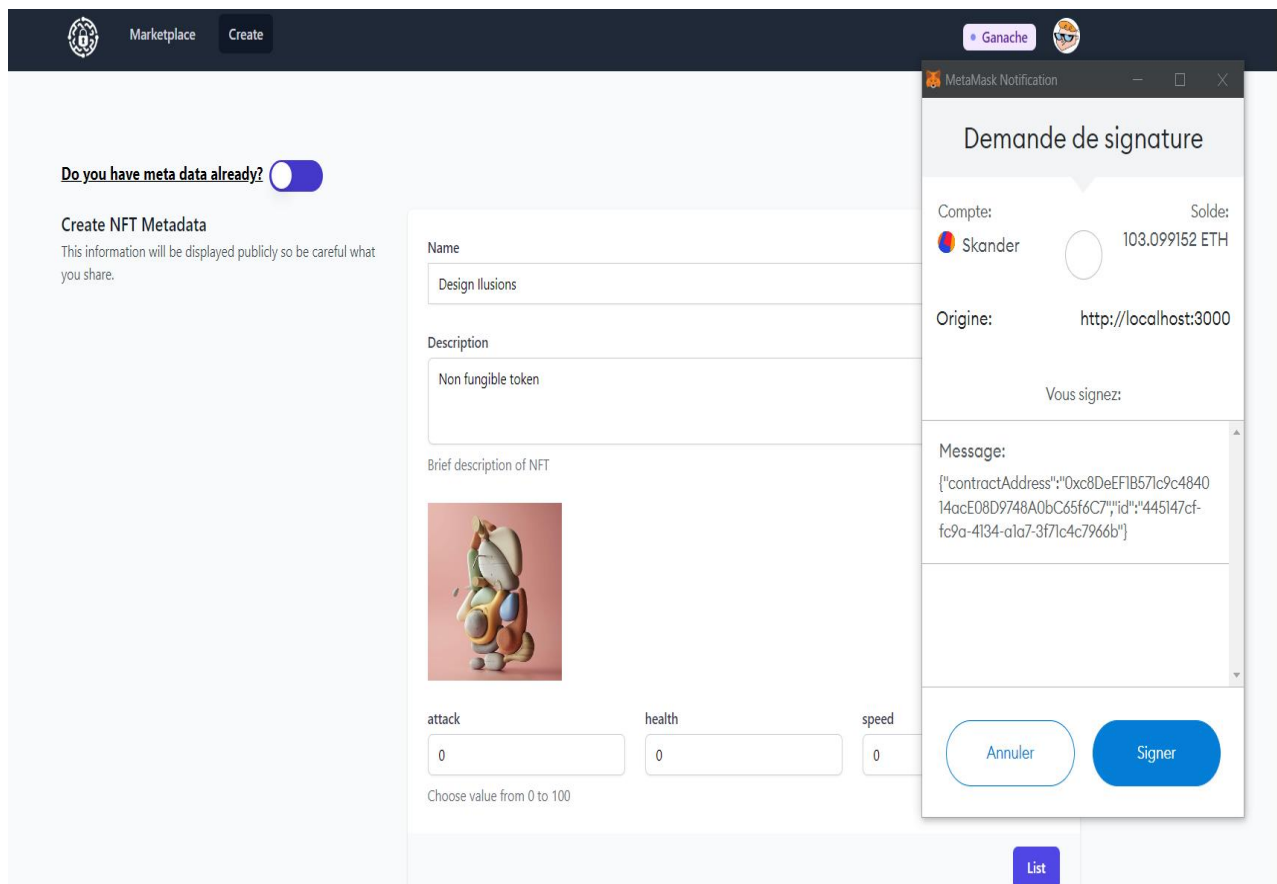


Figure 24 - Signature du NFT

L'utilisateur peut aussi lister un NFT existant via l'interface ci-dessous, il doit juste saisir l'URL du NFT stocké dans le pinata, préciser le prix en ethereum et valider en cliquant sur le bouton « list » une interaction avec le portefeuille de l'utilisateur se produit pour payer le prix de la transaction. Une fois valider le prix saisi par l'utilisateur sera affecté au NFT en question. Comme il est montré dans la figure 25 et 26.

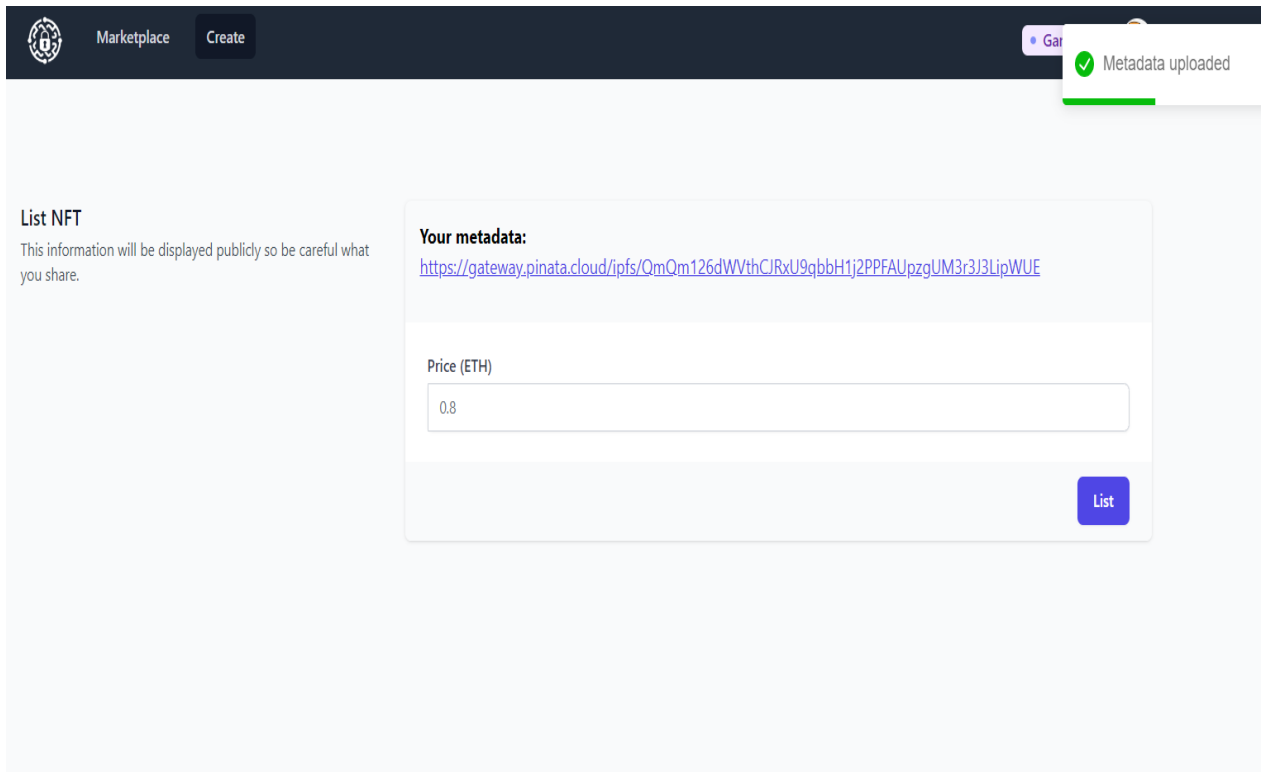


Figure 25 - Interface "Fixer un prix pour un NFT"

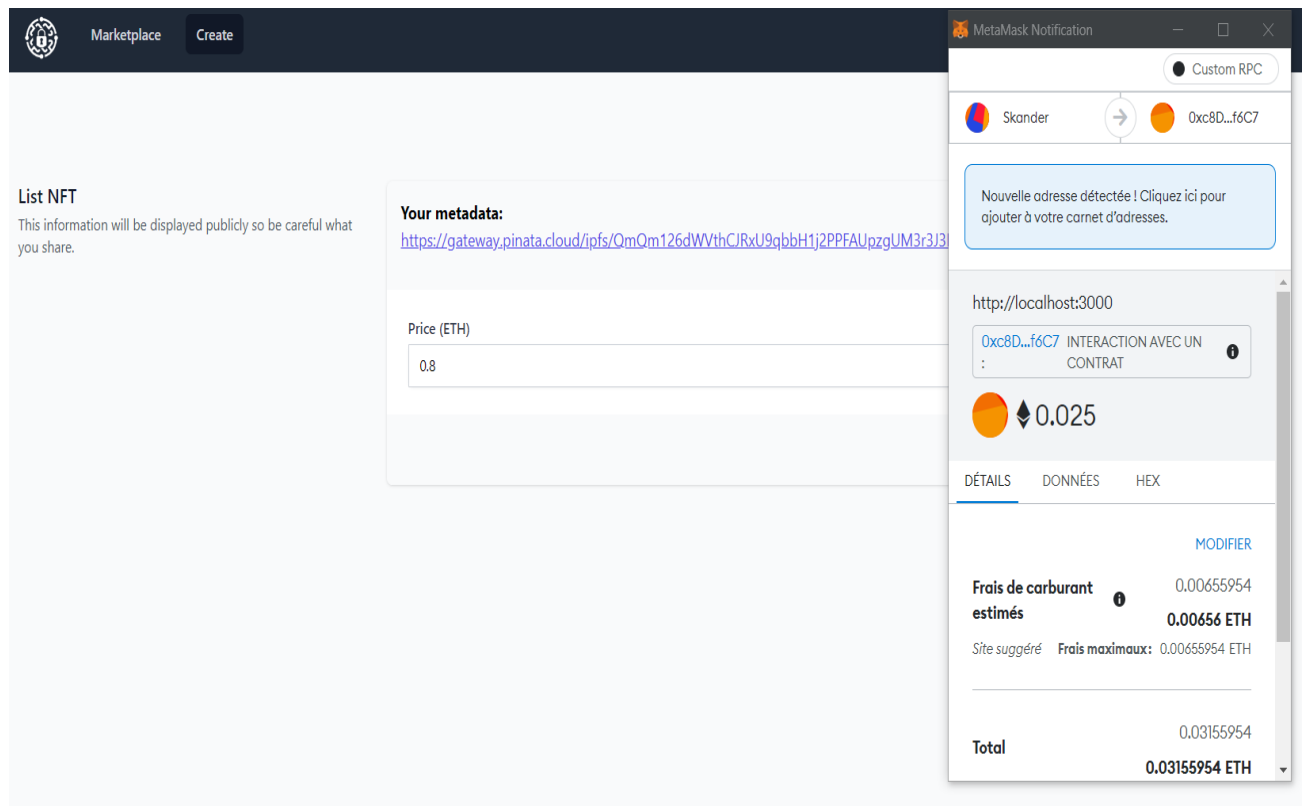
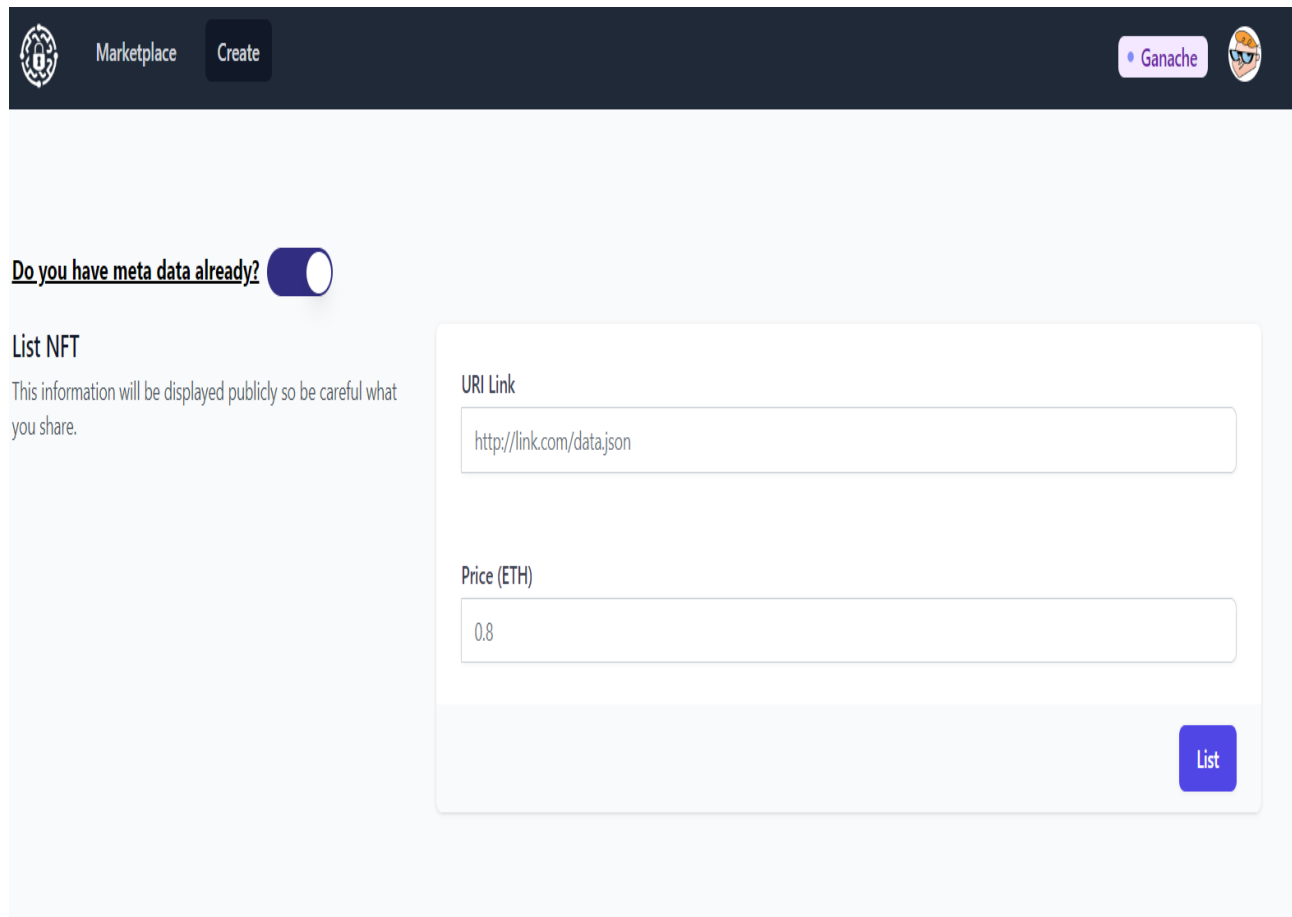


Figure 26- Validation du prix du NFT

La figure ci-dessous permet à l'utilisateur de modifier le prix d'un NFT existant. L'utilisateur doit saisir l'adresse du NFT dans le PINATA, et mettre le nouveau prix du NFT. Une fois validé une interaction avec son portefeuille se produit pour payer le prix de la transaction.



The screenshot shows the 'Marketplace' interface with a 'Create' button. A toggle switch for 'Do you have meta data already?' is turned on. The 'List NFT' section includes a warning: 'This information will be displayed publicly so be careful what you share.' The form contains two input fields: 'URI Link' with the value 'http://link.com/data.json' and 'Price (ETH)' with the value '0.8'. A blue 'List' button is at the bottom right of the form.

Figure 27- Modification du prix d'un NFT existant

5. Diagramme de déploiement

Un diagramme de déploiement est une vue statique qui sert à représenter l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis ainsi que la relation entre eux. Les éléments utilisés par un diagramme de déploiement sont principalement les nœuds, les composants, les associations et les artefacts. Les caractéristiques des ressources matérielles physiques et des supports de communication peuvent être précisées par stéréotype.

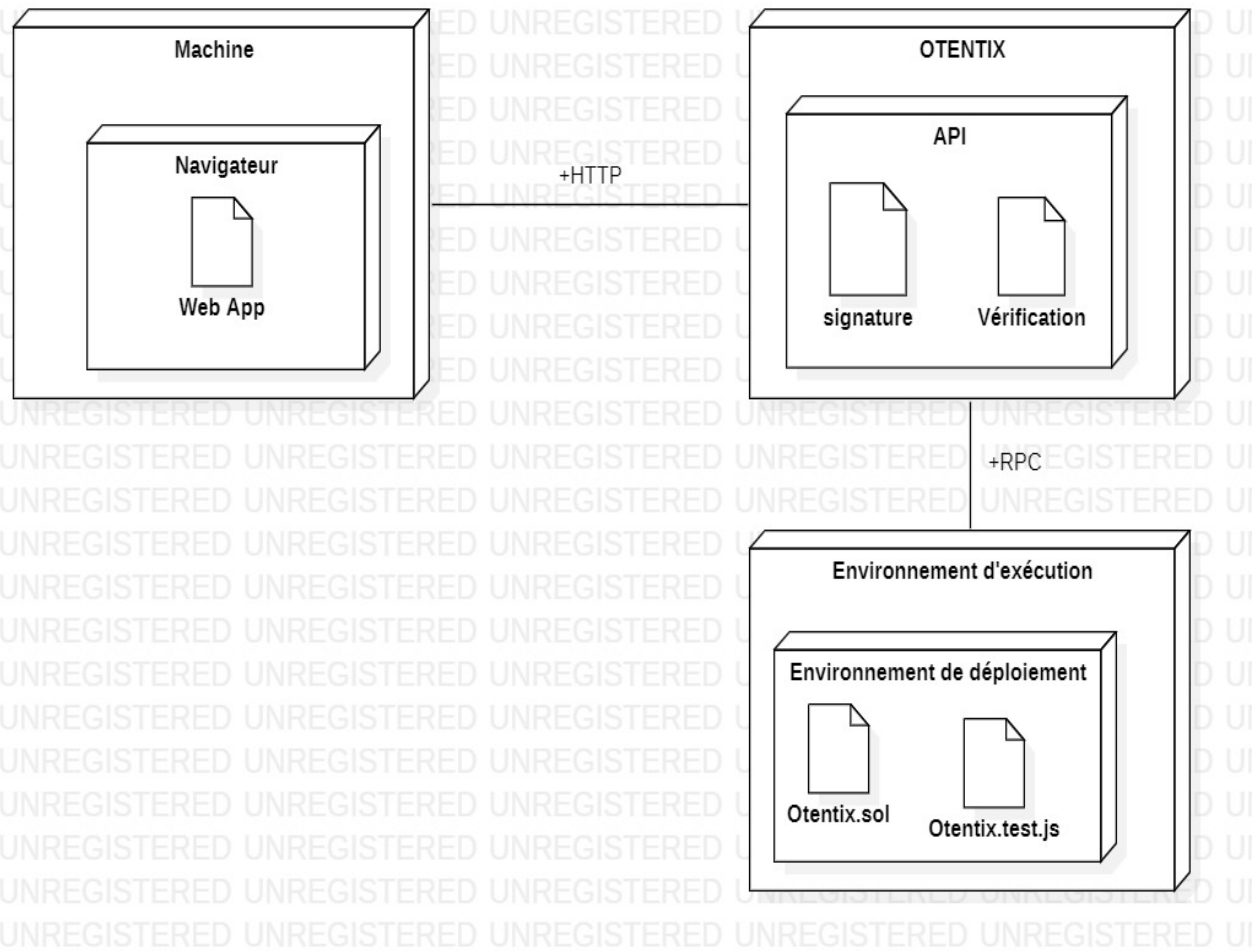


Figure 28 - Diagramme de déploiement

Conclusion

Tout au long de ce chapitre on est arrivé à fournir des outils pour l'analyse, la conception et la mise en œuvre de systèmes logiciels, ainsi que pour la modélisation de processus métier et d'autres processus similaires.

Chapitre 5 : Sprint 3 :

Test et déploiement de l'application

1.Introduction

Après avoir développé les fonctionnalités de l'application, nous passons au troisième sprint qui est le test et le déploiement de l'application. Ce chapitre englobera toutes les étapes de développement de l'application.

2.Test unitaire

Il est très courant d'écrire et de compiler manuellement le code solidity, ce qui convient parfaitement aux petits projets. Cependant, à mesure que notre projet grandit de plus en plus, il est bon d'avoir un moyen automatique de développement du contrat intelligent. De plus, tester le code solidity est crucial pour éviter toute situation problématique causée par un bug dans le contrat intelligent. Truffle en fait partie et est souvent considéré comme le framework Ethereum Swiss Knife car il s'agit d'un environnement de développement, d'un framework de test et d'un pipeline d'actifs pour Ethereum. Voyons maintenant le cycle de développement d'un contrat intelligent avec truffle. Le flux de travail général pour le développement de contrats intelligents avec truffle est réalisé en suivant les étapes suivantes :

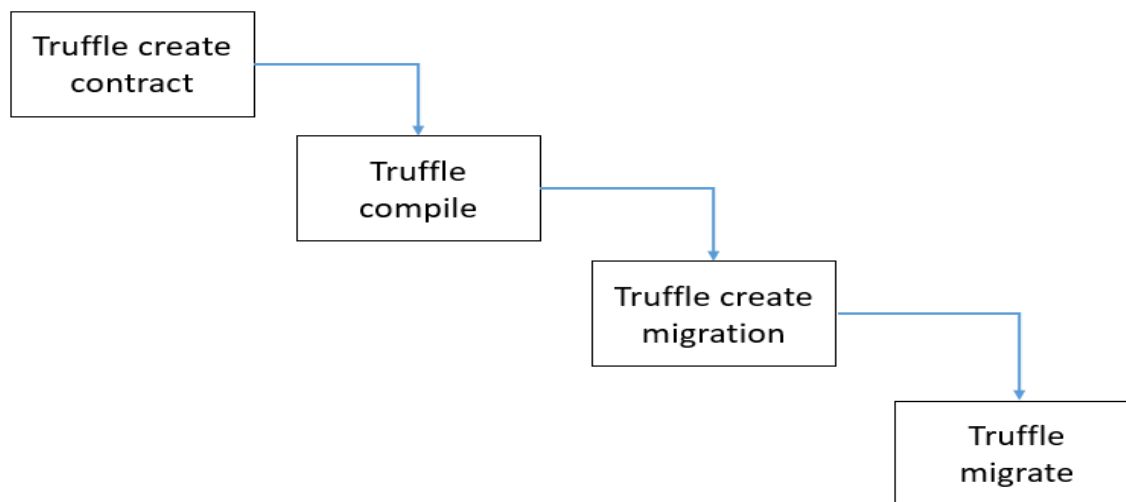


Figure 29 - Cycle de développement du contrat intelligent.

Les deux premières étapes nous les avons découvert durant les deux premiers sprints, alors que la troisième étape consiste à migrer le contrat intelligent.

Afin de déployer un contrat intelligent sur la blockchain, nous devons d'abord écrire les configurations de migration dans un fichier JavaScript. Ces fichiers sont essentiellement responsables de la mise en scène des tâches de déploiement des contrats intelligents sur la blockchain. Mais pour déployer les contrats intelligents avec des migrations, nous devons accéder

à leurs artéfacts qui ont été créés à la suite d'une **truffle compile** commande. Passons maintenant à la phase de test :

En général, le test de contrat intelligent peut être écrit à la fois en solidity et en JavaScript. Dans ce didacticiel, nous continuerons avec JavaScript en utilisant la notation async/wait. Ci-dessous le résultat des tests effectués :

```
MERIEM@DESKTOP-9417N62 MINGW64 ~/Desktop/otentix/nft-otx (QrCode)
$ truffle test
Using network 'development'.

Compiling your contracts...
=====
> Compiling .\contracts\Migrations.sol
> Compiling .\contracts\otentix.sol
> Compiling .\contracts\otentix.sol
> Artifacts written to C:\Users\MERIEM\AppData\Local\Temp\test--16456-VtETLOetYw1k
> Compiled successfully using:
   - solc: 0.8.13+commit.abaa5c0e.Emscripten.clang

Contract: Otentix
  Mint token
    ✓ owner of the first token should be address[0] (122ms)
    ✓ first token should point to the correct tokenURI (105ms)
    ✓ should not be possible to create a NFT with used tokenURI (667ms)
    ✓ should have one listed item (177ms)
    ✓ should have create NFT item (117ms)
  Buy NFT
    ✓ should unlist the item (115ms)
    ✓ should decrease listed items count (101ms)
    ✓ should change the owner (106ms)
  Token transfers
    ✓ should have two NFTs created (128ms)
    ✓ should be able to retrieve nft by index (329ms)
    ✓ should have one listed NFT (143ms)
    ✓ account[1] should have one owned NFT (128ms)
    ✓ account[0] should have one owned NFT (149ms)
  Token transfer to new owner
    ✓ accounts[0] should own 0 tokens (82ms)
    ✓ accounts[1] should own 2 tokens (115ms)
  List an Nft
    ✓ should have two listed items (95ms)
    ✓ should set new listing price (1396ms)

17 passing (11s)
```

Figure 30 – Test

3. Déploiement

Un déploiement est le résultat de la création de votre projet et de sa mise à disposition via une URL en direct.

Lors des déploiements, le projet sera téléchargé et transformé en une sortie prête pour la production grâce à l'utilisation d'une étape de build. Une fois l'étape de build terminée avec succès, un nouveau déploiement immuable sera disponible à l'URL d'aperçu. Les déploiements sont conservés indéfiniment sauf s'ils sont supprimés manuellement.

3.1 Vercel API

Vercel est un service Cloud, dans le périmètre de AWS où Google Cloud Platform, qui permet de déployer des applications.

L'API Vercel peut être utilisée pour effectuer des déploiements en faisant une requête HTTP POST au point de terminaison concerné, y compris les fichiers que vous souhaitez déployer en tant que corps.

3.2 GitHub

Vercel permet des déploiements automatiques sur chaque poussée de branche et fusionne avec la branche de production de vos projets GitHub, GitLab et Bitbucket.

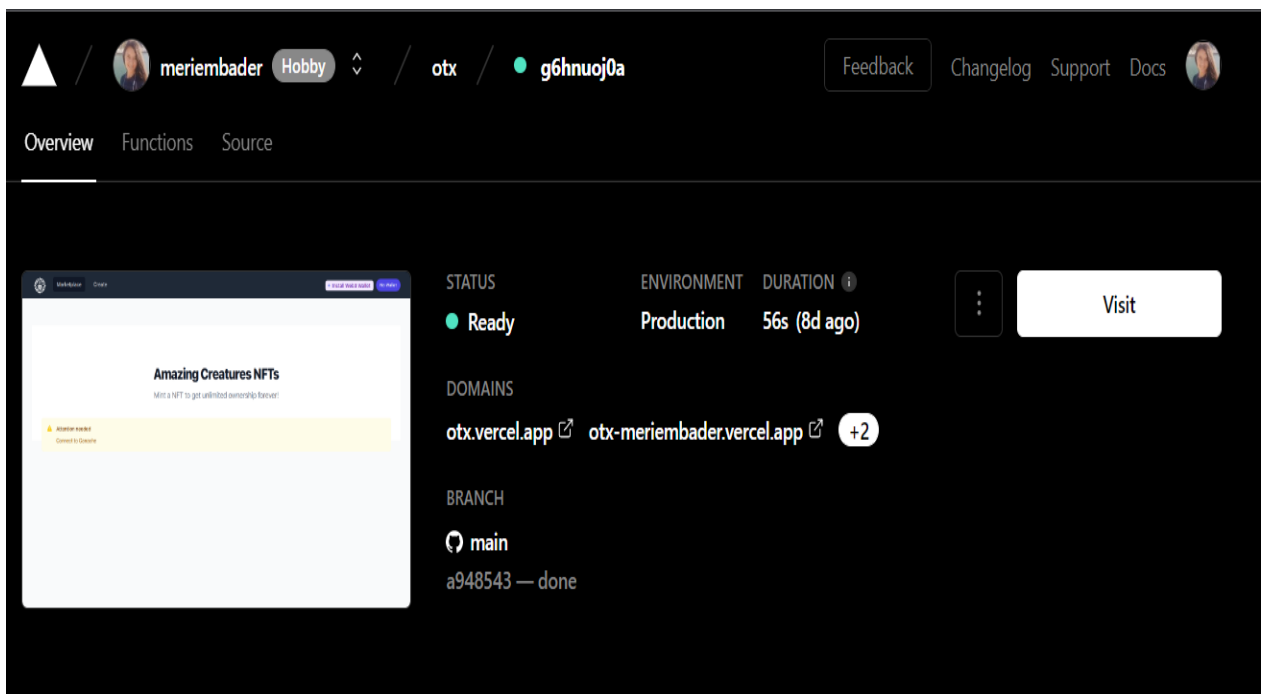


Figure 31- Dashbord Vercel & déploiement Otentix

Notre application Otentix est disponible sous le domaine : <https://otx.vercel.app/>

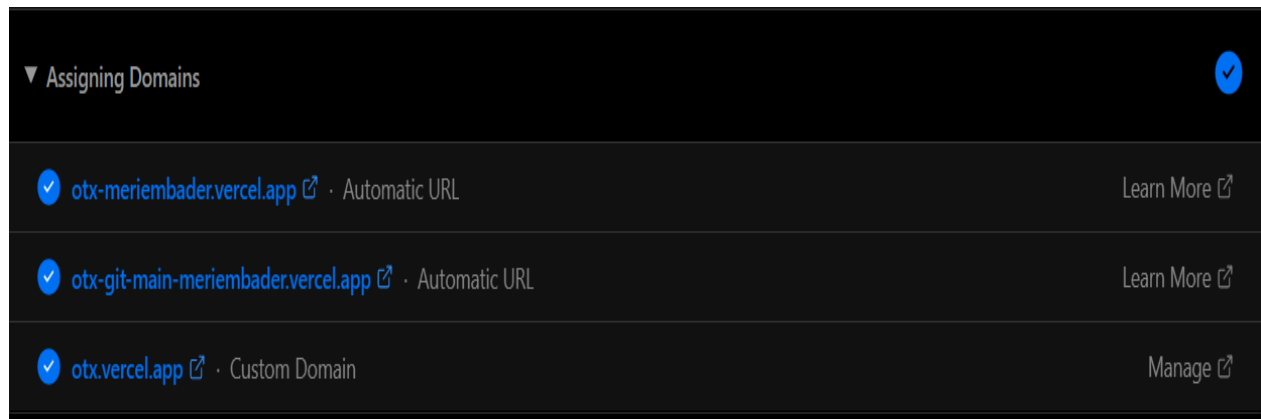


Figure 32- Domaines Otentix

Conclusion

Ce chapitre fût l'occasion de détailler le troisième sprint de l'application. Il nous a permis d'expliquer les étapes de développement du contrat intelligent.

Conclusion générale

Ce projet de fin d'étude réalisé au sein de l'équipe « Direction système d'information » de l'entreprise AGILZ a porté sur la conception et le développement d'une application d'authentification par blockchain qui permet à l'utilisateur de s'authentifier à travers un NFT (jeton non fongible).

L'objectif principal de faire l'authentification par blockchain est de protéger les données personnelles de l'utilisateur. Grâce à notre solution, il va pouvoir s'authentifier sans utiliser ces données.

Pour réaliser notre solution d'authentification par blockchain, nous avons effectué en premier lieu, une étude préalable qui nous a permis de bien comprendre l'objectif de notre projet. Ensuite nous avons fixé les fonctionnalités qui doivent être offertes par notre application. Nous avons par la suite, procédé à la conception de notre application avant de passer à la phase réalisation.

L'application d'authentification par blockchain « OTENTIX » que nous avons développée, offre aux utilisateurs l'opportunité de mieux protéger leurs données personnelles et de s'authentifier en toute sécurité. Dans ce contexte, Agilz bénéficie d'un véritable avantage concurrentiel en mettant en place une nouvelle solution d'authentification.

En dépit de toutes ces fonctionnalités fournies, notre projet peut encore être amélioré, il est ouvert à plein de perspectives qui vont lui donner plus d'ampleur.

Netographie

- [1] <https://agilz.net/> (consulté le 08/04/2022)
- [2] <https://www.simelabs.com/nft-glossary-for-newbies/> (consulté le 20/05/2022)
- [3] <https://fr.reactjs.org/docs/getting-started.html> (consulté le 20/05/2022)
- [4] <https://www.simelabs.com/nft-glossary-for-newbies/> (consulté le 20/06/2022)
- [5] <https://brborghi.tumblr.com/post/121902228754/les-roles-de-scrum/> (consulté le 25/05/2022)
- [6] <https://www.nutcache.com/fr/blog/roles-scrum/> (consulté le 28/05/2022)
- [7] <https://cryptoast.fr/quest-ce-que-solidity-definition-explication/> (consulté le 02/06/2022)
- [8] <https://vercel.com/docs/rest-api> (consulté le 02/06/2022)
- [9] <https://www.lemagit.fr/definition/GitHub> (consulté le 06/06/2022)