



# Développeur Full-Stack

**L3 Miage Sorbonne**

Année universitaire : 2023-2024

**ZHENG Raymond**

Stage réalisé du 7 juin 2024 au 2 juillet 2024 au sein de BC24

Tuteur dans l'organisation: **TAMBONE Quentin**, [quentino.tambonne@gmail.com](mailto:quentino.tambonne@gmail.com)

Tuteur enseignant : **HERBAUT Nicolas**

Membre du jury: **BEN RABAH Nourhène**

## Table des matières

Remerciements	3
Introduction	3
Présentation de l'organisation	4
Missions réalisées en entreprise	6
API Python blockchain	6
WebApp Symfony et implémentation des deux API	8
Les objectifs réussis	10
Stockage de donnée	10
API et anglais	11
Vie professionnel et vie personnel	13
Hardware, SSH et Raspberry Pi	14
Blockchain, MetaData et optimisation du code	18
Intégration API encryptage	20
DataBase	21
Les objectifs ratés / non-complétés	23
Modify resource admin	23
Additional Features	24
Bilan et expérience acquise	25
Références	27

## Remerciements

Pour ce stage de L3 j'aimerais remercier quelques personnes grâce à qui j'ai pu m'améliorer rapidement et prendre conscience de mes problèmes en tant que développeur et qui ont su me guider :

- L'administration de Paris 1 Panthéon-Sorbonne : qui m'a permis de faire ma L3 en MIAGE
- Dr Nicolas HERBAUT : Sponsor du projet et tuteur du stage
- Master 2 : Créateur de l'entreprise
- Master 1 : Gestion et développeur du projet
- Quentin TAMBONE (M1) : Tuteur dans l'entreprise
- Etienne Pascal BAUMGARTNER (M1) : Chef technique / tech lead
- Licence 3 : Développement de l'API python et de l'interface web

## Introduction

Lorsque l'on achète un produit dans un supermarché, une boulangerie, une boucherie ou dans les commerces du quotidien, il est naturel de vouloir connaître sa composition et son origine. C'est ce que propose BC24, une petite entreprise créée par des étudiants de la prestigieuse université Paris 1 Panthéon-Sorbonne. L'entreprise offre une traçabilité complète du produit, permettant de remonter jusqu'aux ingrédients primaires, de savoir d'où ils proviennent, où ils sont transformés (s'il y a transformation), les trajets et modes de transport utilisés, la température, ainsi que tout type d'information pouvant aider le consommateur à mieux comprendre ce qu'il a entre les mains. Il y a une vingtaine de personnes qui travaille autour du projet BC24 (nom du projet et de l'entreprise). Il aboutira à la rédaction de plusieurs papiers/ mémoire de fin d'études basés sur le projet. Notre sponsor, le Dr Nicolas HERBAUT, est à l'origine de l'idée. Ensuite, un groupe de Master 2 en IKSEM (parcours anglais de la formation MIAGE) s'occupe de l'administration du projet et de l'évaluation de sa faisabilité. Les étudiants de Master 1 jouent un rôle crucial en gestion de projet et développement, de plus ils encadrent notamment trois équipes de Licence 3 et un stagiaire. Les groupes de Licence 3 de MIAGE sont divisées en deux groupes, la première travaillant sur la création d'une API en Python et un groupe sur le développement d'une interface web. Pendant ce stage, j'étais sous la supervision des Master 1, les aidant dans l'implémentation de leurs projets, l'API blockchain ainsi que dans les deux autres projets des équipes de Licence 3, afin de respecter le cahier des charges établi par les étudiants de Master 2. Nous commencerons donc par une présentation plus détaillée de l'entreprise et de son organisation. Ensuite, nous examinerons les missions que j'ai pu accomplir pendant ce stage et les difficultés rencontrés pendant ce stage. Enfin nous conclurons par un bilan sur ce que j'ai pu apprendre, tant du côté professionnel que du côté humain, que j'ai pu tirer de cette expérience.

## Présentation de l'organisation

Dans un monde où il est de plus en plus facile d'acheter tout et n'importe quoi, n'importe où, les produits peuvent provenir de n'importe quelle région du monde. Si une personne souhaite, par exemple, limiter son empreinte carbone en achetant uniquement des produits issus de l'Union européenne, il peut être difficile de trouver des informations précises sur la provenance des ingrédients utilisés dans les produits qu'elle achète. De même, pour une personne soucieuse du bien-être animal, savoir si un animal a été élevé en plein air ou dans une ferme d'élevage intensif peut être un critère important.

C'est là qu'intervient BC24, une application web en partenariat avec des entreprises pour faciliter la traçabilité des produits. Simple et rapide, elle permet de trouver facilement toutes sortes d'informations liées à un produit, telles que : la ferme d'élevage, le transport de l'animal ou de l'aliment, la température de transport (pour s'assurer du respect de la chaîne du froid), l'usine d'abattage ou de transformation, etc. Ces informations sont précieuses pour certains consommateurs, et BC24 fournit un service adapté à ces besoins spécifiques.

Bien qu'il existe déjà des marchés et des applications similaires, BC24 se distingue par son approche innovante : l'utilisation de smart contracts<sup>1</sup>. Grâce aux smart contracts sur la blockchain Ethereum<sup>1</sup>, nous pouvons garantir la fiabilité des informations, grâce à la transparence et à l'immutabilité que permet la blockchain. Comme le souligne CoinHouse (2024) : « C'est cette blockchain qui rend les transactions transparentes, infalsifiables, et irréversibles... Ils simplifient et fluidifient les transactions entre différents acteurs, qui n'ont pas besoin de se faire confiance ni même de se connaître, et ont donc toutes les chances de s'imposer. Non seulement en raison de leurs avantages (sécurité, réduction des risques d'erreur, suppression des frictions liées aux obligations contractuelles, etc.) ». Ces avantages et la confiance accordée à la blockchain ont poussé BC24 à choisir les smart contracts comme base de projet pour la traçabilité.

L'entreprise BC24 regroupe une vingtaine de personnes travaillant ou ayant travaillé sur le projet. Tout d'abord, nous avons :

- Dr Nicolas Herbaut : Sponsor du projet et également la personne ayant eu l'idée du projet. Il nous fournit le matériel nécessaire à la réussite du projet, avec le soutien de l'université Paris 1 Panthéon-Sorbonne.
- Master 2<sup>2</sup> : Un groupe de 4 étudiants en IKSEM, responsables de la création de l'entreprise. Ils s'occupent de la gestion de l'entreprise et définissent les caractéristiques du produit que nous devons livrer à la fin du projet.
- Master 2 bis : Deux étudiants qui réalisent leurs mémoires de fin d'études sur le projet BC24 / sur une partie du projet.
- Master 1<sup>3</sup> : Un groupe de 5 étudiants en alternance, tous très polyvalent. Ils ont pour mission de manager le projet, de comprendre les demandes des étudiants de Master 2 pour les traduire et les transmettre aux étudiants de Licence 3. Ils sont responsables de la partie

---

<sup>1</sup> Une cryptomonnaie

<sup>2</sup> Florent, Alix, Katia

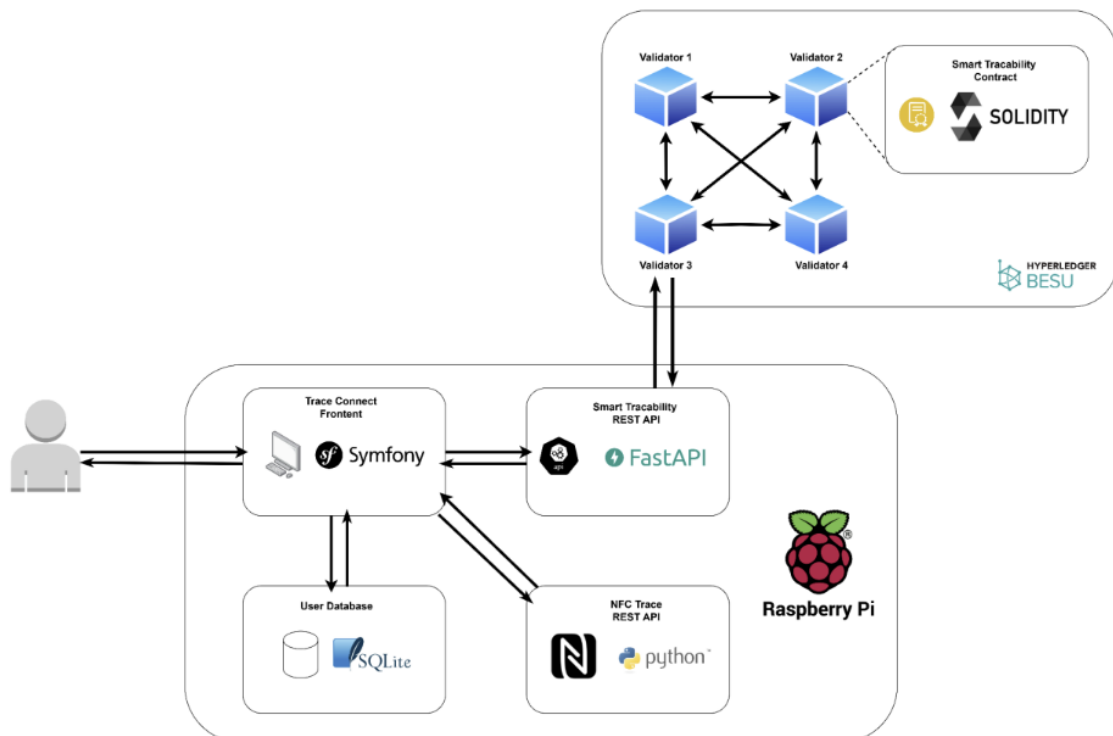
<sup>3</sup> Paul-Cesar, Chadi, Etienne, Quentin, Hugo

hardware du projet, ainsi que de toute la partie liée à la blockchain. Ce groupe est le pivot central du projet BC24.

- Licences 3 : Le plus grand groupe, qui peut être divisé en deux plus petits groupes :
  1. Le premier groupe NFC-Trace <sup>4</sup> composé eux aussi de deux groupes, l'un de cinq élèves de la classique de la MIAGE et le second de quatre élèves alternants qui ont repris le projet NFC-Trace à la fin pour finir cette partie du projet car il manquait encore certaines fonctionnalités. Ce projet est l'écriture d'information crypté sur le tag NFC puis la lecture de celle-ci via l'utilisation d'une API Python.
  2. Le second groupe Trace-Connect<sup>5</sup> composé aussi de quatre étudiants de la filière classique de la Licence 3 de la MIAGE était le projet centrale, créer un site web compatible avec les 2 autres projets et qui par la suite devra implémenter l'API Python sur l'écriture NFC-Trace des Licences 3 et l'API blockchain Smart-Traceability du groupe des Master 1.
- Stagiaire / mission d'entreprise, c'est donc le poste que j'occupe pendant mon stage, mon travail consiste donc à implémenter les 2 API et les faire fonctionner correctement ensemble avec l'aide des Master 1 encore présent sur le projet.

Donc ma position est une position de développeur en soutien aux développeurs de Master 1 qui connaissent moins bien le projet Symfony développé par le groupe numéro deux de Licences 3 de la classique MIAGE.

Voici donc comment fonctionne/interagisse les projet entre eux



<sup>4</sup> Nino, Adrian, Aymeric, Angel, Khera, Elyahou, Alex, Omar, Katia

<sup>5</sup> Dan, Raymond, Seonho, Make

Nous pouvons donc voir 2 blocs qui communiquent entre eux, le premier en haut à droite, le côté blockchain, utilisant la blockchain de l'Ethereum et des smart contracts, on peut facilement interagir avec la blockchain sans rien y connaître grâce à l'API Smart Tracability qui est située dans le second bloc qui est donc lancé par le Raspberry Pi, cette API blockchain communique donc avec notre WebApp Trace Connect qui est notre site web développé en Symfony sur lequel j'ai déjà et vais travailler, ensuite nous avons le second API de cryptage de données, NFC Trace qui sert à écrire et lire de la donnée cryptée, puis nous avons en plus de la blockchain une base de données, développée initialement avec SQLite mais qui sera changée par la suite et enfin l'utilisateur qui peut accéder à tous ces modules via l'application web qui fait donc le lien et qui est le pivot central du projet.

## Missions réalisées en entreprise

### API Python blockchain

Ma première mission consistait à reprendre l'API Python blockchain<sup>6</sup> sur laquelle travaillaient les Master 1 au même moment. Le projet n'étant pas encore terminé du côté de l'API, il fallait finaliser l'API côté blockchain pour pouvoir accomplir la seconde mission, qui était la plus importante. En binôme avec Étienne, membre du groupe de Master 1, nous étions chargés de terminer les fonctionnalités au plus vite. C'était un projet crucial, car c'était la seule porte pour communiquer avec la blockchain que nous avions à notre disposition, donc il était impératif de le terminer pour ensuite passer au plus gros du projet : la refonte du projet web en Symfony<sup>7</sup>.

Pour cette première mission, nous avons cinq ou six fonctions à finaliser, avec un squelette déjà préparé. Le travail consistait à compléter le squelette avec un algorithme efficace, de sorte que les fonctions ne prennent pas trop de temps à s'exécuter et soient optimisées, tout en respectant certaines demandes et certains critères de temps qui nous étaient imposés.

Après avoir discuté avec Quentin (tuteur de stage), nous avons prévu d'y consacrer une semaine, ce qui nous laisserait encore deux à trois semaines pour finir la refonte du côté web, ce qui nous semblait faisable à première vue. J'ai donc commencé par consulter de la documentation sur la bibliothèque web3.py<sup>8</sup> de Python sur Internet. Cette bibliothèque permettait de comprendre comment fonctionne le web3 et comment coder en utilisant les smart contracts fournis par les Master 1, que j'utilisais pour coder en Python. L'objectif était de comprendre un maximum de choses durant les deux premiers jours, de tester le plus de code possible, de comprendre les erreurs et d'obtenir un maximum de feedback de la part des Master 1 pour voir ce que je faisais bien ou mal. Ensuite, j'ai enchaîné les jours restants de la semaine en essayant de finaliser deux fonctionnalités par jour, en gardant le vendredi pour corriger tous les bugs et les cas de figure que j'aurais pu oublier ou laisser de côté, dans le but d'avoir le code le plus propre possible et de valider, si possible, la première mission le vendredi soir. (Voir mission ci-dessous)

---

<sup>6</sup> [https://github.com/bc24-miage-dev/BC24-API/blob/master/api/routes/roles\\_routes.py](https://github.com/bc24-miage-dev/BC24-API/blob/master/api/routes/roles_routes.py)

<sup>7</sup> Framework PHP, <https://symfony.com/>

<sup>8</sup> Bibliothèque Python pour communiquer avec la blockchain

- api skeleton ✓
- Github structure ✓
- docker image for production ✓
- authentication of the user that calls the rest api / API Access token per user (walletAdress in POST request) ✓
- Wallet address and private keys can be hold in seperate file (.env or json) ✓
- Raymond pi : ssh pi@176.177.37.154 ✓

Ce furent trois jours très compliqués, à tel point que nous avons finalement décidé d'abandonner cette mission et qu'Étienne<sup>9</sup> continuerait seul car ce serait plus rapide. Au début, j'étais très motivé, mais j'ai essayé pas mal d'échecs lors de mes premières tentatives, ne comprenant pas ce qu'étaient une transaction, une wallet address, etc. Je me suis rapidement retrouvé bloqué en codant, essayant de comprendre les termes techniques utilisés dans la documentation. Il m'est arrivé de chercher une fonctionnalité pour laquelle il était nécessaire d'utiliser un certain paramètre que j'ai dû rechercher sur Internet pour comprendre à quoi il faisait référence, puis ce paramètre contenait lui-même d'autres paramètres que je ne comprenais pas non plus. Cela devenait rapidement infernal, et je me perdais avec tous ces termes que je ne comprenais pas tout à fait. J'ai ainsi passé deux journées entières à lire de la documentation, à regarder des vidéos en ligne, principalement sur YouTube, et à tester du code en vain. Je pensais qu'en regardant des vidéos avec des exemples que je pouvais recopier, cela m'aiderait à progresser dans ma compréhension du sujet. Cela m'a certes grandement aidé pour comprendre certains aspects, mais cela ne m'aiderait pas sur le sujet principal : je ne comprenais pas comment fonctionnait la blockchain. Je pense avoir essayé d'avancer trop rapidement, en sautant trop d'étapes au début, et j'ai négligé ce qui était le plus important, à savoir les bases : qu'est-ce que la blockchain, en quoi est-elle utile dans ce projet, comment l'utiliser, quels sont les prérequis... Évidemment, j'ai essayé de demander de l'aide, notamment à Étienne, probablement la personne la plus expérimentée en tant que développeur dans l'équipe, mais il ne pouvait pas être tout le temps disponible : parfois en cours, parfois en période de partiels, et parfois en entreprise. Même s'il me répondait rapidement, cela restait compliqué. Après deux ou trois jours sans trop avancer, c'est à ce moment qu'ils ont décidé qu'il était préférable de me retirer de cette mission et de laisser Étienne seul sur l'API pour une meilleure efficacité et répartition du travail.

D'un point de vue extérieur, on pourrait dire que ce n'était qu'une perte de temps, car finalement on m'a retiré de cette mission, et j'ai dû passer deux journées entières sur quelque chose que je n'ai pas réutilisé dans le projet. Cependant, ce n'est pas tout à fait le cas. Je pense que cette expérience, bien que catastrophique et probablement l'une des phases les plus compliquées de mon stage, m'a permis de grandir en termes de compréhension, en tant que développeur et en tant que personne. Elle m'a donné une vision plus large de ce que faisaient les Master 1 sur la blockchain. Même si je n'ai compris que partiellement comment utiliser le web3 et la blockchain, cela m'a aussi permis d'apprendre à utiliser les API REST<sup>10</sup>, à construire une API. En développement web, on utilise souvent les requêtes POST et GET, par exemple, mais j'ai pu comprendre plus précisément à quoi correspond un POST et un GET, et dans quels cas il est préférable d'utiliser l'un ou l'autre, que ce soit

---

<sup>9</sup> Membre du groupe de Master 1 et mon référent pour les questions blockchain

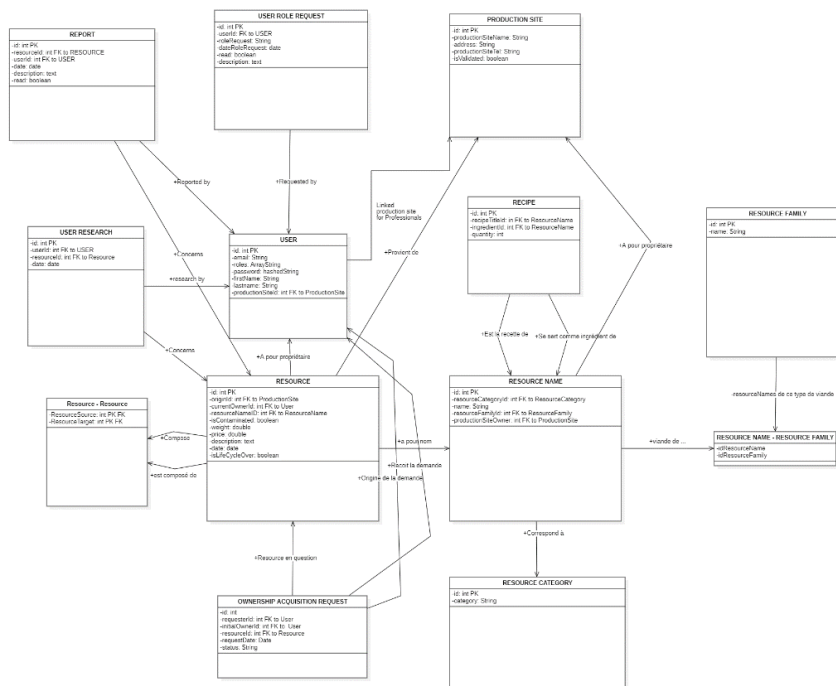
<sup>10</sup> Une catégorie d'API

pour coder ou pour les appels d'API. Surtout, j'ai appris à utiliser FastAPI<sup>11</sup>, qui est une bibliothèque très répandue pour la création d'API en Python. Il est très probable que je sois amené à réutiliser la bibliothèque FastAPI dans un avenir proche.

## WebApp Symfony et implémentation des deux API

Pour cette partie, la WebApp est constituée de 10 « requirements <sup>12</sup>», dont 7 doivent être réalisés par moi-même. Au début, il n'y avait pas autant de requirements, mais au fur et à mesure que le projet devenait de plus en plus concret, et que nous approchions du rendu final, nous nous sommes rendu compte, surtout Quentin qui est mon maître de stage, qu'il manquait pas mal de choses, d'où l'augmentation de la charge de travail et du nombre de requirements.

Mais tout d'abord, regardons ce qu'est la WebApp. Il s'agit d'une application web développée en Symfony par le groupe de Licence 3 classique dont je faisais partie avant le stage pour valider deux matières : « Technologies du Web : Niveau Avancé » et « Projet commun ou concours ». Ce groupe est composé de 4 étudiants : Dan KLECZEWSKI (chef de projet, full-stack), Seonho KIM (full-stack), Make SAGMA (développeur front), et Raymond ZHENG (full-stack). Nous avons pour objectif de développer une application web en Symfony (Trace-Connect) pour le back-end et d'utiliser Twig pour le front-end, avec une base de données en SQLite. Notre mission était de respecter au mieux les fonctionnalités définies par les Master 1, tout en respectant les critères de notation donnés par nos professeurs de « Technologies du Web : Niveau Avancé » et « Projet commun ou concours », d'où l'intégration d'une base de données centralisant toutes les ressources (voir ci-dessous).



<sup>11</sup> Bibliothèque Python pour créer des API REST

<sup>12</sup> <https://www.notion.so/Internship-Integration-to-webapp-e22860e484f14af58913e85f8606d886>



Développée à l'origine pour tout type d'aliment, nous avons dû nous concentrer sur une catégorie précise, à savoir les viandes, en raison de contraintes et de demandes des Master 2. Cela a entraîné l'ajout de plusieurs fonctionnalités, telles que le signalement d'une ressource contaminée, la création de recettes par les usines qui transforment la viande, un système de demande de rôle, et tout un système de gestion des utilisateurs via les comptes admin. Certaines de ces fonctionnalités seront réimplantées dans le projet BC24.

La WebApp étant le pivot central du projet BC24, et donc la partie la plus importante du projet, et ayant grandement contribué à la réalisation du projet « Trace-Connect », ma mission principale était d'implémenter les deux API (blockchain et écriture/lecture de tags<sup>13</sup>) tout en respectant les requirements des Master 1.

Pour cette partie de la mission, il me restait donc deux semaines et demie. En ce qui concerne l'organisation, nous avons prévu de réaliser le requirement 3 en une semaine, puis d'utiliser la semaine restante pour compléter les autres requirements, plus simples et rapides à réaliser, avant de finir avec la correction des bugs pendant le dernier week-end et la fin de la semaine. Nous avons prévu d'utiliser Notion pour une meilleure gestion de projet, où tous les requirements seraient stockés. Un autre Notion était dédié à l'utilisation de tickets, ce qui me permettrait d'obtenir plus de précisions sur un sujet ou sur la manière dont ils souhaitaient le réaliser. Enfin, des réunions étaient prévues tous les soirs (cours en journée ou travail en entreprise) ou tous les deux jours pour suivre l'avancement et obtenir des retours en vocal pour plus de précision. Nous avons convenu que, malgré les nombreux tickets, la meilleure approche était de tenir des réunions vocales aussi souvent que possible pour faciliter la communication et éviter toute erreur d'interprétation des demandes.

---

<sup>13</sup> <https://github.com/bc24-miage-dev/BC24-NFC-Trace>

# Les objectifs réussis

## Stockage de donnée

```
REQUIREMENT 3 : Implement the following creation of resource UI/UX and Integration -- in order to keep consistency
1. https://127.0.0.1:8000/pro/eleveur/naissance :
+ :: Remove the 'id' textfield at the bottom ✓
+ :: When I click on "Confirmer la naissance" -> creation of the animal NFT on the blockchain -> opening of a modal asking me to "present an NFC tag near the scanner, then click on the write button to associate the new NFT created" and display an "Ecrire" button -> When I press the "Ecrire" button, launch the writing of the NFT_tokenID on the NFC tag detected -> If successful, then success message, otherwise failed to write to nfc, then error message ✓
2. https://127.0.0.1:8000/pro/equarrisseur/equarrir/{id}
+ :: Remove the textfield to insert the NFC tag ✓
+ :: When I click on "Valider" -> creation of the carcasse NFT on the blockchain -> opening of a modal asking me to "present an NFC tag near the scanner, then click on the write button to associate the new NFT created" and display an "Ecrire" button -> When I press the "Ecrire" button, launch the writing of the NFT_tokenID on the NFC tag detected -> If successful, then success message, otherwise failed to write to nfc, then error message ✓
3. https://127.0.0.1:8000/pro/equarrisseur/decoupe/{id}
+ :: When I click on "Découper 1/2" -> creation of the first demi-carcasse NFT on the blockchain -> opening of a modal asking me to "present an NFC tag near the scanner, then click on the write button to associate the new NFT created" and display an "Ecrire" button -> When I press the "Ecrire" button, launch the writing of the NFT_tokenID on the NFC tag detected -> If successful, then success message + disable "Découper 1/2" button, otherwise failed to write to nfc, then error message ✓
+ :: Same to be done for the button "Découper 2/2" ✓
4. https://127.0.0.1:8000/pro/usine/decoupe/{id}
+ :: When I click on "Valider" on each line of the tab -> creation of the morceau NFT on the blockchain -> opening of a modal asking me to "present an NFC tag near the scanner, then click on the write button to associate the new NFT created" and display an "Ecrire" button -> When I press the "Ecrire" button, launch the writing of the NFT_tokenID on the NFC tag detected -> If successful, then success message + disable the "Valider" button associated to the line, otherwise failed to write to nfc, then error message. ✓
5. https://127.0.0.1:8000/admin/add
+ :: Remove the "id" textfield at the bottom
+ :: When I click on "Add Resource" -> creation of the NFT (breeder, slaughterer, manufacturer) on the blockchain if resource needed are available -> opening of a modal asking me to "present an NFC tag near the scanner, then click on the write button to associate the new NFT created" and display an "Ecrire" button -> When I press the "Ecrire" button, launch the writing of the NFT_tokenID on the NFC tag detected -> If successful, then success message + refresh the page with empty inputs, otherwise failed to write to nfc, then error message.
⚠ Call needed : Call the /XXX route in the blockchain API to create a new resource and retrieve the resource's NFT_tokenID. Call the /write route of the hardware API by passing the NFT_tokenID, in order to set also the NFC tag associated to the resource created
```

Comme vous pouvez le voir pour le requirement 3, il y a beaucoup de travail, même si cela peut sembler répétitif. Ayant commencé en pensant que cela prendrait « seulement » une semaine, j’ai rapidement déchanté, avec une charge de travail de 8 à 10 heures par jour durant la première semaine (week-end compris).

Les premiers jours ont surtout été consacrés à la compréhension et à l’implémentation. Comme vous pouvez le voir dans le diagramme UML de la page 7, presque toutes les tables sont reliées à la table ‘resources’, qui est la table la plus importante. Cependant, avec l’implémentation de l’API blockchain, nous devons revoir l’arrangement des tables et supprimer certains liens pour assurer une cohérence dans la base de données. Ce n’était pas de tout repos. La première étape consistait à examiner le requirement 3 et à identifier toutes les classes du programme et de la base de données que je devais supprimer. Étant donné que le projet est en PHP orienté objet, nous utilisons des classes pour représenter les tables grâce à Doctrine, ce qui est très pratique pour gagner du temps, en plus de l’utilisation de Symfony qui facilite les commandes avec Doctrine et génère automatiquement les tables et les relations entre elles. Cependant, il est plus compliqué de modifier les relations après leur création. C’était là le premier gros problème que j’ai rencontré : comment supprimer des entités de tables, des relations, et d’autres informations inutiles sans casser la base de données.

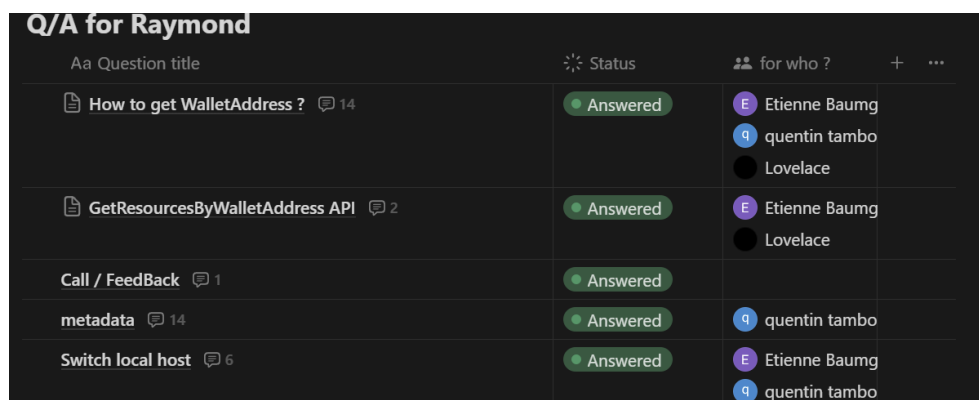
Il est simple de construire une base de données avec Doctrine et Symfony, mais beaucoup plus difficile de la réorganiser. C'était un des points sur lesquels j'ai dû passer le plus de temps, car les problèmes de base de données sont, d'expérience, parmi les plus compliqués à résoudre. Nous avons passé des soirées entières à résoudre des problèmes de Doctrine non compatibles avec la base de données, parce que nous avons modifié quelque chose dans la base de données sans passer par Doctrine<sup>14</sup> et les migrations<sup>15</sup>, ce qui faisait planter le système lors des migrations. Ne voulant plus revivre cela, j'ai fait pas mal de recherches sur des sites et regardé des vidéos pour comprendre comment fonctionnait Doctrine, jusqu'à pouvoir écrire les tables sans utiliser les commandes mises à disposition.

Cependant, cela prenait beaucoup de temps. Le plus sûr de mon côté était donc de faire plusieurs sauvegardes grâce à Git, de cloner plusieurs versions de la base de données, puis de faire des modifications tout en conservant les anciennes versions dans un dossier.

## API et anglais

Travaillant principalement avec Étienne, le développeur principal de l'API blockchain avec qui j'étais en contact, je me retrouvais souvent à lui parler pour améliorer l'API sur laquelle il travaillait, demander l'ajout d'une fonctionnalité, ou obtenir des conseils sur la manière d'utiliser certaines parties de l'API, et sur ce que je devais lui renvoyer pour que l'API accepte correctement ma demande. Il faut prendre en compte que l'API n'étant pas encore terminée, je devais m'adapter à Étienne, tout comme il devait s'adapter à moi, ce qui nous a pas mal rapprochés.

En ce qui concerne mes demandes et questions liées au projet, j'avais initialement pensé les adresser sur le serveur Discord, en créant un canal spécialement dédié à cet effet. L'idée était de conserver une trace écrite de mes problèmes et interrogations. Cependant, nous avons rapidement constaté que cette méthode devenait ingérable. C'est à ce moment-là que nous avons décidé de passer à l'utilisation de Notion avec des tickets pour un meilleur suivi. Cette solution s'est avérée bien plus structurée que Discord. Notion a apporté une réponse adéquate à nos besoins, tant en termes de suivi que de rapidité de réponse. Certaines personnes ayant désactivé les notifications du serveur Discord, elles ne recevaient pas toujours les questions, alors qu'avec Notion, tout était plus simple et centralisé. (Voir ci-dessous)



Aa Question title	Status	for who?
How to get WalletAddress ? 14	Answered	Etienne Baumg, quentin tambo, Lovelace
GetResourcesByWalletAddress API 2	Answered	Etienne Baumg, Lovelace
Call / FeedBack 1	Answered	
metadata 14	Answered	quentin tambo
Switch local host 6	Answered	Etienne Baumg, quentin tambo

<sup>14</sup> ORM qui sert de mapping dans Symfony

<sup>15</sup> Migration de base de données, qui nous permet tous d'avoir la même base de donnée

Ce qui m’a parfois ralenti dans mon travail était que certaines fonctionnalités n’étaient pas encore développées ou ne fonctionnaient pas correctement (ce qui n’était pas de leur faute). Il m’est donc arrivé de me retrouver bloqué, incapable d’avancer sur le requirement. Dans ces cas-là, je choisissais souvent de passer à un autre requirement et de créer un ticket pour éviter de perdre du temps en attendant simplement qu’on règle mon problème. Je pense que cette initiative, bien que naturelle, était aussi l’une des plus importantes, car je travaillais en décalé par rapport aux Master (le soir après leurs cours ou alternance pour eux, et pour moi le midi ou l’après-midi). J’ai souvent dû attendre des réponses, parfois le matin, qui n’étaient fournies qu’à la pause de midi ou résolues le soir, ce qui aurait pu me faire perdre beaucoup de temps.

Enfin, nous avons parfois rencontré un problème que je pense retrouver en entreprise à l’avenir : la communication avec Étienne. N’étant arrivé en France que récemment, il n’est pas encore parfaitement bilingue, même s’il comprend la plupart des conversations. Nos discussions étaient souvent un mélange d’anglais et de français. L’un des cas les plus mémorables concernait l’utilisation du mot « retrieve » dans le contexte suivant :

```
REQUIEREMENT 7 : Displaying the resource list of the owner
Every place in the webapp where you can retrieve the list of resources owned :
• Resource list for all role
So :
1. Call the /XXX route of the blockchain API to retrieve the resources linked to the walletAddress
2. Display the resource list as we already do
```

Je l’avais simplement traduit par : 1. Appel le chemin /XXX de l’API blockchain et retirer les ressources liées à la « walletAddress ». Se traduisant tout d’abord par récupérer via WorldReference<sup>16</sup> je pensais qu’il faisait allusion au consommateur qui souhaitais acheter un des aliments donc le retirer de la circulation et de notre Système d’information, donc le considérer comme étant utilisé / autre terme désignant le fait qu’il est arrivé en fin de vie et donc qu’on n’aura pas plus d’information sur lui. Ce problème de traduction qui est totalement ma faute, m’aura embêté une journée entière, j’ai embêté Etienne une journée pour lui demander d’implémenter/changer une fonctionnalité, mais heureusement en fin d’après-midi nous avons pu éclaircir le malentendu.

Principales traductions		
Anglais		Français
<b>retrieve [sth] ⇒ vtr</b>	(get, find: physical object)	récupérer ⇒ <i>vtr</i> (chien) aller chercher ⇒ <i>vtr</i>
	Bill threw the stick and the dog retrieved it. <i>Bill a lancé le bâton et son chien est allé le chercher.</i>	
<b>retrieve [sth] vtr</b>	(get, find: computer files, data) (Informatique)	extraire ⇒, récupérer ⇒ <i>vtr</i>
	Linda hoped she would be able to retrieve her files after her computer crashed. <i>Linda espérait pouvoir extraire (or: récupérer) ses dossiers après que son ordinateur ait planté</i>	

<sup>16</sup> <https://www.wordreference.com/enfr/retrieve>

Ce que je retiens c'est qu'il faut attention à la traduction possible, ne pas simplement regarder la première traduction la seconde traduction peut-être la bonne et s'il y a un doute demander à la personne concernée pour éclaircir et éviter tout malentendu.

## **Vie professionnel et vie personnel**

Avec seulement trois semaines pour effectuer le stage et réaliser la mission, la gestion du projet et du temps consacré à chaque partie était cruciale pour ne pas prendre de retard et livrer le projet à temps. Il est vrai que, les projets prennent du retard pour diverses raisons dans les entreprises. En France, « tous secteurs confondus, un quart des projets dépasseraient les fameuses deadlines fixées initialement » (LeMagIT). Dans notre cas, il était impératif de respecter les délais, car la soutenance ne pouvait être déplacée. C'est dans ce contexte que j'ai dû accomplir mon stage et ma mission dans un temps réduit.

L'une des choses que je souhaitais pour mon premier stage était de pouvoir travailler physiquement avec les membres de l'équipe. Malheureusement, les circonstances ne l'ont pas permis. Toutefois, le problème ne résidait pas tant dans le fait de ne pas voir mes collègues en personne. Avec les outils modernes comme Internet, Zoom, et le télétravail, cela est devenu un défi pour de nombreuses entreprises, même si cela ne semblait pas particulièrement nous affecter. Ce qui m'a vraiment posé problème, c'était le décalage entre mes horaires de travail et ceux des Masters. Travaillant de 9h30 à 17h, suivant les horaires des magistes (9h30-12h30 puis 14h-17h), nous avons décidé de suivre plus ou moins le même planning pour la première semaine. Cependant, j'ai rapidement remarqué que cela posait plusieurs problèmes.

Pendant que je travaillais, les Masters étaient soit en cours, soit en entreprise, ce qui les empêchait de répondre à mes questions. Et lorsque j'avais officiellement terminé ma journée, ils commençaient à travailler sur le projet après leurs cours. Cette situation d'horaires opposés nous a beaucoup impacté lors notre première semaine de travail. Cela soulevait un gros problème : que faire si je me retrouvais bloqué ? Au début, j'ai pu contourner cet obstacle en travaillant sur un autre requirement, ce qui semblait logique. Mais cette approche a ses limites, surtout lorsqu'on est bloqué dès le début sur un élément fondamental qui impacte tous les autres requirements. C'était ma plus grande problématique au début du stage, une difficulté qui ne survient généralement que dans les entreprises opérant à l'international.

Ce problème, qui semblait insoluble au début, s'est finalement résolu plus ou moins naturellement. En travaillant en début de journée, parfois je restais bloqué et avec le temps j'ai commencé à me coucher de plus en plus tard en attendant des réponses pour avancer sur le projet. Il m'est ainsi arrivé de me coucher plusieurs fois à 5h ou 6h du matin pour finir une fonctionnalité ou corriger des bugs. Le plus souvent, je me couchais vers 1h ou 2h du matin, car les Masters, travaillant sur le projet plutôt en soirée, étant plus disponibles à ces heures-là. Cela me permettait d'avoir un flux d'échanges continu, ce qui facilitait grandement l'avancement du projet. Ces échanges m'aidaient aussi à mieux comprendre le fonctionnement de la blockchain et de l'API, ce qui me permettait de mieux m'adapter et mieux voir les possibilités que m'accordais l'API.

Ainsi, au milieu/à la fin de la première semaine, mes horaires se sont modifiés, le travail de 9h30 à 17h se transforma en 13h-18h, puis 20h-1h ou plus. Bien que cela fût plus pratique pour la communication, cela a également entraîné des problèmes. En me couchant tard, je me levais tard et ne répondais pas vraiment aux messages le matin. Je pense que les Masters ont pu croire que je ne travaillais pas le matin, ce qui a pu créer des complications. Cependant, voyant que je rendais le travail à peu près dans les délais et que je continuais à avancer, il n'y a pas eu de répercussions notables.

Je ne saurais dire si c'était la meilleure décision. Il est évident que ce n'est pas une pratique viable en entreprise et que ce n'est pas la solution idéale. Mais, manquant de temps et voyant que cette méthode semblait plus efficace que de travailler en décalé, j'ai décidé de maintenir ces horaires jusqu'à la fin du stage. Heureusement, cela n'a pas posé de problèmes aux Masters, qui ont accepté cette organisation.

### **Hardware, SSH et Raspberry Pi**

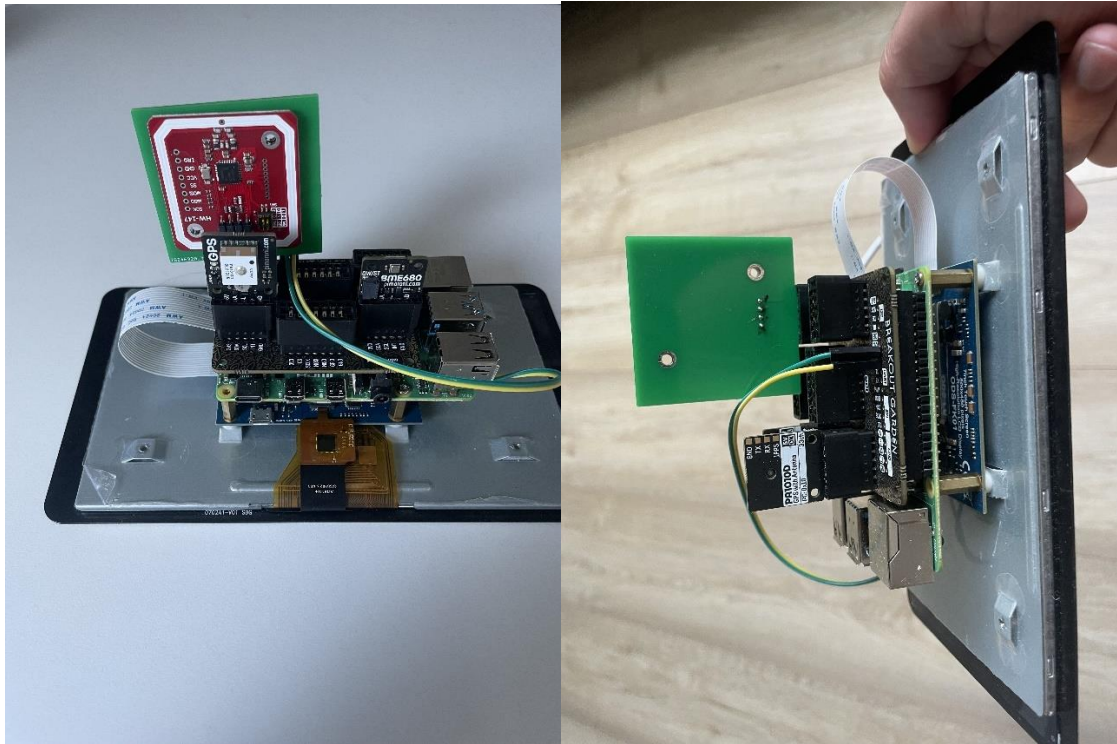
« Le Raspberry Pi n'est rien d'autre qu'un ordinateur réduit à sa plus simple expression : une unique carte à processeur ARM un poil plus grande qu'une carte de crédit » (Clubic). Voilà comment est souvent décrit un Raspberry Pi, un petit ordinateur facile à transporter, « abordable et accessible à tous » (Clubic). Utilisant le modèle B avec 8 Go de RAM, ce Raspberry Pi<sup>17</sup> était censé être la pièce maîtresse du projet physique. Il devait contenir tous les projets et permettre de lancer l'API blockchain en local avec une connexion réseau pour les interactions avec la blockchain, de faire fonctionner l'API de cryptage en local<sup>18</sup>, de lire et écrire les informations du tag, et enfin de lancer le projet web Symfony avec une connexion pour accéder à la base de données.

Bien que cet usage ne soit qu'un exemple de ce qui est possible, il est évident qu'une partie de ces applications étaient destinées à être hébergées sur le web si le projet venait à être déployé un jour, plutôt que de fonctionner uniquement en local.

---

<sup>17</sup> <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>

<sup>18</sup> Application lancée en local



Comme vous pouvez le voir (ci-dessus), le Raspberry Pi (en vert en dessous) est équipé d'un "hat" permettant de brancher une carte SD et un scanner <sup>19</sup>(vert et rouge) qui sert à lire des tags. Nous y trouvons également un capteur de température et un capteur GPS intégrés, conçus pour prévenir les fraudes et les erreurs concernant la température ou la provenance des produits. Grâce au programme informatique, tout est automatisé pour le relevé de température et la localisation GPS. Enfin, vous pouvez apercevoir, posé contre la table, un petit écran utilisé pour l'interface graphique.

Pour cette partie du projet, nous avons mis en place une connexion SSH <sup>20</sup> afin de rendre le Raspberry Pi accessible aux autres membres pour effectuer des tests. « SSH signifie "Secure Shell", il s'agit donc d'un "shell" dit "sécurisé" [...] Un shell va donc nous permettre d'administrer nos serveurs Linux, en local, c'est-à-dire lorsque l'on se trouve physiquement en face de notre serveur, mais aussi à distance, notamment grâce à Secure Shell (SSH) » (IT-Connect). Nous avons également configuré une connexion via VNC Server et VNC Viewer<sup>21</sup>, très pratique pour se connecter à distance au Raspberry Pi. Cette solution a été principalement utilisée pour effectuer des tests et vérifier si le code fonctionnait correctement sur le Raspberry via GitHub.

Mais cela ne s'est pas fait en un clin d'œil. Pour permettre aux autres de se connecter à mon SSH, il a d'abord fallu ouvrir un port sur ma box Internet, redirigeant ainsi vers le Raspberry Pi. Ce n'était pas très intuitif, et n'ayant que peu de connaissances sur le sujet, j'ai dû m'informer sur de nombreux sites. Bien que j'aurais pu demander de l'aide aux masters, j'ai préféré explorer mes options par moi-même pour ne pas les déranger inutilement. C'est ainsi que j'ai consulté plusieurs

---

<sup>19</sup> Permet de scanner un tag NFC et potentiellement voir ce qu'il contient

<sup>20</sup> Secure Shell

<sup>21</sup> Logiciel permettant à des personnes possédant le logiciel de manipuler un ordinateur tiers à distance

articles sur des sites comme IT-Connect ou encore le site de Bouygues, mon fournisseur d'accès Internet.

Après plusieurs heures sans succès, j'ai fini par contacter l'assistance téléphonique. Après une brève explication de mon problème, la personne a ouvert plusieurs ports, du 20 au 25, redirigeant tous vers mon Raspberry Pi. Cependant, ce n'était pas la meilleure option, car certains de ces ports sont vulnérables, et les laisser ouverts présente des risques :

- Port 20 : « File Transfer Protocol – Data » (speedguide, Port 20 Details)
- Port 21 : « File Transfer Protocol » (lonos)
- Port 22 : « Secure Shell » (blog.netwrix)
- Port 23 : « Telnet » (blog.netwrix)
- Port 25 : « Simple Mail Transfer Protocol » (blog.netwrix)

Laisser ces ports ouverts est potentiellement dangereux, car cela peut exposer le système à des attaques par brute force, et ainsi percer à jour les identifiants de la machine ce qui est évidemment très dangereux. C'est pourquoi certains ports sont normalement fermés/ gérés par le pare-feu. « les ports ouverts offrent une 'surface d'attaque' plus étendue ou agrandissent la fenêtre d'opportunités pour un attaquant de trouver des vulnérabilités, des exploits, des mauvaises configurations [...] De plus, les protocoles en texte clair et non chiffrés peuvent conduire à l'espionnage du réseau. » (Specopssoft).

Cependant, n'ayant pas prêté suffisamment attention à ces dangers et ne comprenant pas encore tous les risques associés à l'ouverture de ces ports, je me suis rapidement retrouvé victime d'un piratage. Avec un mot de passe simple de 4 caractères sur le Raspberry et tous les ports de ma box redirigeant vers un Raspberry Pi peu protégé, l'accès a été facile pour l'attaquant. J'ai d'abord remarqué des anomalies : le Raspberry Pi redémarrait parfois tout seul, l'écran s'éteignait et se rallumait pour aucune raison. Ces signes auraient dû me mettre la puce à l'oreille, mais étant connecté en SSH avec le groupe et via VNC Viewer et Server, je savais qu'ils passaient souvent sur le Raspberry pour voir l'avancé, j'ai simplement supposé qu'un membre du groupe faisait des manipulations à distance, comme une mise à jour ou l'installation d'un logiciel.

Le lendemain, je me suis retrouvé avec un Raspberry Pi complètement hors service : plus de connexion Internet, une interface graphique partiellement fonctionnelle avec des fonctionnalités manquantes/ supprimées. J'ai rapidement compris la situation lorsque, après avoir demandé la veille si quelqu'un avait fait des manipulations sur le Raspberry Pi, personne ne m'a répondu. Ce problème m'a coûté une journée entière de travail pour tenter de récupérer une connexion et une interface graphique fonctionnelle, mais à ce moment-là, il n'y avait plus grand-chose qui fonctionnait et je n'arrivais plus à naviguer sur la machine sauf en passant par les commandes linux. (voir image ci-dessous)



```
[ OK ] Started Create Static Device Nodes in /dev.
[FAILED] Failed to start Journal Service.
See 'systemctl status systemd-journald.service' for details.
[ C ] Stopped Journal Service.
Starting Journal Service...
Starting systemd Journal Service...
[ C ] Started Set the console loglevel to debug.
[ C ] Started systemd Colloping all Devices.
Starting Helper to synchronize boot up for ifupdown...
[ OK ] Reached target Local File Systems (Pre).
[ 5:46:59] systemd-journald[157]: Assertion 'clock_gettime(clock_id, &ts) == 0' failed at ../src/basic/time-util.c:55: Function not(). Starting.
[FAILED] Failed to start Journal Service.
See 'systemctl status systemd-journald.service' for details.
[ C ] Stopped Journal Service.
[FAILED] Failed to start Journal Service.
See 'systemctl status systemd-journald.service' for details.
[ C ] Started Helper to synchronize boot up for ifupdown.
[ C ] Started systemd Journal Service.
Starting Show Plymouth Boot Screen...

[ 56.407618] systemd-logind[998]: Assertion 'clock_gettime(clock_id(&clock_id), &ts) == 0' failed at ../src/basic/time-util.c:55: Function not(). Starting.
[ 56.499026] systemd-logind[998]: Assertion 'clock_gettime(clock_id(&clock_id), &ts) == 0' failed at ../src/basic/time-util.c:55: Function not(). Starting.
My IP address is 192.168.1.22 2004:0da1:2b07:8711:90f5:4fa1:9733
Raspbian GNU/Linux 11 pi 11gpi
pi login: pi (custom@pi)
Linux pi 5.10.102-071-rpi320 SMP Tue Mar 8 12:24:00 GMT 2022; armv7l
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Mar 15 08:19:13 CEST 2024 on tty1
vncserver-111620@vnc1: Connection: Cannot find a running X server on vti
```

Il semblerait que le service systemd-journald ait été supprimé. « systemd-journald est un service système qui collecte et stocke les données de journalisation. Il crée et maintient des journaux structurés et indexés basés sur les informations de journalisation reçues de diverses sources. » (freedesktop). On découvrira grâce à monsieur Herbaut que cette suppression visait probablement à effacer les traces du pirate. J'ai discuté de la situation avec les masters, mais ils n'avaient pas de solution immédiate. En début de soirée, nous avons donc organisé un appel d'urgence pour résoudre ce problème. Malgré nos efforts, nous n'avons réussi qu'à rétablir une connexion Internet mais qui ne marchait pas toujours s'activant une fois sur deux au redémarrage.

C'est alors que nous avons décidé de contacter monsieur Herbaut pour obtenir de l'aide. Il a rapidement répondu à notre appel et a su résoudre rapidement le problème en une heure et demie. Il nous a donc annoncé que : nous avons bel et bien été victimes d'un piratage, apparemment par des hackers situés en Chine. Ils avaient effectué des centaines de milliers de tentatives de connexion et avaient obtenu des clés d'accès administrateur une fois entrés dans le système. Heureusement pour nous, Monsieur Herbaut a pu réinstaller une interface graphique fonctionnelle, changer les mots de passe, retirer les clés d'accès administrateur, et fermer les ports Internet vulnérables pour éviter que cela ne se reproduise. Cette journée a probablement été la plus stressante de tout le projet pour moi. Heureusement, la réactivité des masters et de notre sponsor a rapidement permis de contenir la situation, et nous avons pu passer à autre chose.

Cet incident m'a rappelé les dangers d'Internet. À force de passer nos journées en ligne, nous oublions souvent l'importance de bien nous protéger, de choisir des mots de passe sécurisés, et de surveiller nos activités en ligne. Ce rappel brutal à la réalité a été bénéfique pour tout le groupe. Nous avons également découvert que le second Raspberry Pi lui aussi avait subi des milliers de tentatives de connexion, heureusement pour nous, il n'avait pas l'air d'y avoir de dégât comparable au mien.

## Blockchain, MetaData et optimisation du code

Lorsqu'on enregistre la naissance d'un animal, nous collectons des données telles que le lieu de naissance, la date et l'heure, le sexe, etc. Toutes ces informations sont stockées dans un paramètre lié à un token de la blockchain que nous appelons MetaData dans notre projet. Quand il s'agit de rechercher une information ou d'en ajouter une nouvelle, nous passons souvent par la MetaData. En tant que développeur full-stack du projet, mon rôle était de veiller à ce que les bonnes MetaData soient insérées au bon endroit, en utilisant de manière appropriée l'API blockchain fournie par les Masters. Ma mission consistait donc à insérer toutes les données utiles et à les récupérer au bon moment pour les traiter et les afficher correctement sur le front-end de la WebApp.

Étant débordé par le travail à ce moment-là et manquant de temps, j'ai eu l'idée d'insérer toutes les données disponibles, même si certaines données n'étaient pas disponibles, je les remplaçais par 'FALSE', 'NULL' ou 'NAN', selon le type de données. Bien que cette approche semble pratique au premier abord, et je pense qu'elle l'était effectivement, plus le projet avançait et demandait l'ajout de nouvelles données, plus la liste de MetaData s'allongeait. Après l'avoir récupéré via l'API blockchain elle était récupérée via un format JSON puis transformée en tableau, plus facile à manipuler en PHP. Dans le paramètre MetaData, on pouvait retrouver près d'une trentaine de paramètres, ce qui est beaucoup pour un être humain, mais peu pour une machine qui stocke un tableau ou un dictionnaire de 30 paramètres. Avec une indexation rapide, il n'était pas nécessaire d'optimiser cela au début.

Cependant, un problème est rapidement apparu. Tandis que le Raspberry Pi pouvait suivre la cadence de toutes les MetaData, ce n'était pas le cas pour la blockchain. Malheureusement, nous utilisions une blockchain gratuite située au Québec, ce qui entraînait une latence non négligeable dans les réponses de l'API blockchain. Cela devenait problématique car le temps de réponse dépassait largement une seconde lorsqu'on souhaitait écrire des informations sur la blockchain. Ce problème technique a persisté longtemps. Nous en avons beaucoup discuté avec les Masters, en essayant de review le code pour voir les optimisations possibles afin de minimiser les écritures sur la blockchain. Toutefois, cela était inévitable : le but de notre projet étant la traçabilité, limiter les écritures aurait compromis la base même du projet qui est d'avoir le plus d'information possible.

Après de multiples tests, nous nous sommes rendu compte que le principal problème résidait dans l'écriture et la récursivité du code. Bien que la récursivité soit généralement utile pour optimiser, elle nous faisait ici défaut. J'ai donc tenté de réduire au maximum les appels à l'API, en stockant autant de variables que possible entre les pages pour éviter de redemander des informations à la blockchain, mais cela restait inévitable.

Après une review du code plus approfondi, nous nous sommes dit qu'il fallait couper la MetaData et insérer que les paramètres nécessaires pour chaque aliment et chaque étape de production, on avait donc ceci (voir ci-dessous, photo de gauche), mais avec un nombre de paramètre multiplié par 2 ou plus :

```

public function metadataTemplate(array $information): array {

    $templateArray = [
        'isContaminated' => false,
        'disease' => null,
        'weight' => null,
        'price' => null,
        'description' => null,
        'genre' => null,
        'address' => null,
        'birthPlace' => null,
        'birthDate' => null,
        'nutrition' => null,
        'vaccin' => null,
        'approvalNumberBreeder' => null,
        //-----Animal-----//
        'slaughteringPlace' => null,
        'carcassDate' => null,
        'slaughtererCountry' => null,
        'approvalNumberSlaughterer' => null,
        //-----Carcass-----//
        'demiCarcassDate' => null,
        //-----DemiCarcass-----//
        'manufacturingPlace' => null,
        'meatDate' => null,
        'manufacturingCountry' => null,
        'approvalNumberManufacturer' => null,
        //-----Manufacturing-----//
        'transportDate1' => null,
        'transportDate2' => null,
        'travelTime' => null,
        //-----Transport-----//
    ];
    $mergedMetaData = array_merge($templateArray, $information);
    return $mergedMetaData;
}

public function metadataTemplateMeat(array $information) : array {
    $templateArray = [
        'isContaminated' => false,
        'manufacturingPlace' => null,
        'meatDate' => null,
        'manufacturingCountry' => null,
        'approvalNumberManufacturer' => null,
        'transportDate1' => null,
        'transportDate2' => null,
        'travelTime' => null,
        "gpsStart" => null,
        "gpsEnd" => null,
        "temperatureStart" => null,
        "temperatureEnd" => null,
    ];
    $mergedMetaData = array_merge($templateArray, $information);
    return $mergedMetaData;
}

```

On peut voir ici qu'une partie des MetaData est inutile pour certaines ressources. Par exemple, pour une carcasse, il n'est pas nécessaire de savoir si elle a été vaccinée, et les informations nutritionnelles n'ont pas de sens. De même, pour un morceau de viande, certains paramètres ne sont pas pertinents. Cependant, pour des raisons de rapidité et de simplicité, j'avais décidé à ce moment-là de tout regrouper pour développer le moins de méthode possible et réduire le nombre de fonction à utiliser.

La solution à ce problème a donc été de segmenter au maximum les MetaData. J'ai dû créer un modèle spécifique de MetaData pour chaque type de ressource. Cela a allongé le temps de développement, mais a rendu le système plus intuitif et mieux organisé. Le problème était que j'ai dû dupliquer une grande partie des fonctions, qui étaient presque identiques, mais différaient légèrement les unes des autres. J'avais aussi envisagé de structurer le projet en utilisant la programmation orientée objet avec du polymorphisme. Cette approche prometteuse était plus complexe à mettre en place et, par manque de temps, j'ai opté pour une solution plus rapide, bien que moins efficace et moins pratique pour quelqu'un qui reprendrait le projet après moi. Si j'avais eu plus de temps, je pense que j'aurais priorisé la refactorisation de cette partie.

Avec cette nouvelle approche, nous avons réussi à réduire de moitié, voire plus, la quantité de MetaData pour certaines ressources (voir image de droite ci-dessus). Cependant, la courbe de performance n'était pas spécialement linéaire. Les tests effectués par Étienne et Paul-César ont révélé des lenteurs du côté de l'API blockchain, particulièrement lors des écritures, qui étaient beaucoup plus longues que les lectures. Ces délais dépendent également de la taille de la blockchain. En effet, plus il y a d'informations stockées, plus la recherche, la modification, et l'écriture sont lentes. Pour lire la blockchain, chaque bloc contenant des informations doit être parcouru, sachant qu'un bloc est créé toutes les 5 secondes. Au bout de quelques jours, on se retrouve donc avec une quantité astronomique de blocs d'informations, ce qui nous obligeait souvent à effacer toutes les données. Cela pouvait être gênant, car il m'arrivait de tester quelque chose un jour, puis de revenir le

lendemain pour découvrir que cela ne fonctionnait plus. J'ai ensuite compris que l'information était restée dans le cache de l'application web, et que lorsque je tentais de récupérer cette information via la blockchain pour obtenir plus de détails, une erreur était renvoyée dû au fait qu'elle n'existait plus à ce moment-là.

## **Intégration API cryptage**

Après avoir réussi à structurer les MetaData, à réduire au maximum la latence entre les requêtes de l'API, à optimiser mon code pour minimiser les appels à l'API, et à résoudre les problèmes rencontrés avec le Raspberry Pi, nous étions enfin prêts à passer à l'étape suivante : l'écriture et la lecture de NFT sur le tag NFC. L'objectif était d'inscrire l'identifiant du NFT de la blockchain dans une partie du tag, puis de pouvoir scanner le tag pour retrouver les informations associées via l'application web. Pour cela, nous devons également intégrer l'API d'encryptage développée par les étudiants en Licence 3, en formation classique et en alternance.

Malheureusement, tout ne s'est pas déroulé comme prévu. Le premier groupe n'a pas réussi à terminer le projet à temps. Sur les trois tâches principales — l'écriture, la lecture et l'encryptage des données pour garantir la sécurité — ils n'ont complété qu'une partie de la première. Ils avaient rejoint le projet un mois avant la date de rendu et n'ont donc pas pu beaucoup avancer. Quant au second groupe, ils ont abandonné le projet en cours de route, mais ont tout de même rendu un travail partiel, mais il manquait toujours la partie cryptée du projet. Face à cette situation, les Masters ont décidé de renoncer à l'idée de crypter les données stockées dans le tag.

Nous nous retrouvions donc à devoir implémenter une API incomplète, mais avec les fonctionnalités les plus cruciales. Avant mon arrivée, Quentin avait déjà commencé à implémenter la lecture du tag NFC. Mon objectif était donc de lire le tag, récupérer les informations, puis les stocker dans un champ de saisie.

Cependant, la tâche s'est révélée plus complexe que prévu. Je ne disposais pas de tout le matériel nécessaire. J'avais seulement le "hat" permettant de lire et d'écrire sur le tag, contrairement au groupe des Masters qui disposait de modules supplémentaires comme le capteur de température et le GPS. J'ai dû modifier manuellement une partie du code pour pouvoir démarrer le programme. Un autre obstacle majeur était le manque de documentation sur le projet. Les alternants étant en entreprise et le projet officiellement rendu, il ne restait plus personne pour répondre à mes questions. J'ai donc dû lire et comprendre le code existant pour en saisir le fonctionnement.

Contrairement à l'API blockchain que je codais en PHP, principalement en utilisant le back-end et les URL fournies par les contrôleurs, j'étais ici confronté à un problème différent. Il fallait pouvoir scanner plusieurs fois un tag sur la même page, ce qui m'empêchait de changer d'URL à chaque appel de l'API, comme je le faisais habituellement. Je me suis retrouvé coincé, et c'est à ce moment-là que je suis allé demander conseil aux Masters. Ils m'ont orienté vers l'utilisation de JavaScript. Le problème, c'est que j'avais très peu d'expérience en JavaScript et que je n'avais jamais fait d'appels d'API avec ce langage. De plus, l'API ne fonctionnait qu'en local, et j'avais du mal à l'utiliser, contrairement à l'API blockchain, pour laquelle il existait une documentation et des personnes à qui je pouvais demander de l'aide.

J'ai passé toute une journée sans succès, essayant de passer par Symfony pour ensuite rediriger vers du JavaScript, mais en vain. Après avoir regardé des dizaines de sites et vidéos, sur les conseils de Dan Kleczewski<sup>22</sup>, qui faisait du JavaScript en stage à ce moment-là, je lui ai demandé de l'aide. Il m'a heureusement recommandé une vidéo<sup>23</sup> de 12 heures qui couvrait les bases du JavaScript. Finalement, après avoir abandonné l'idée de passer par PHP, et après une nuit de réflexion, j'ai réalisé qu'il était possible d'utiliser PHP pour appeler une URL grâce à un contrôleur, qui à son tour appellerait l'API pour me renvoyer un JSON avec les bonnes données. Avec ces données, je pourrais écrire correctement les informations dans le tag NFC.

Cela a pris du temps à mettre en place, mais ma deuxième journée a été beaucoup plus productive. Après plus de 15 heures bloqué sur ce sujet, j'ai enfin pu résoudre le problème en passant par le front-end. Les Masters, dont un membre qui était spécialisé en développement web, n'étaient pas vraiment satisfaits de cette approche. J'ai compris par la suite que ce n'était pas une pratique recommandée, car cela pouvait poser des problèmes de CORS (Cross-Origin Resource Sharing). Comme l'explique la documentation : « Les erreurs Cross-Origin Resource Sharing (CORS) se produisent lorsqu'un serveur ne renvoie pas les en-têtes HTTP requis par la norme CORS. Pour corriger une erreur CORS provenant d'une API REST ou d'une API HTTP de la passerelle API Gateway » (repost).

## DataBase

L'une des dernières tâches du projet consistait à effectuer la migration vers une base de données en ligne. Avec la collaboration de notre sponsor, qui nous a fourni un accès à une base de données en ligne MariaDB, et l'aide précieuse de Chady, un développeur full-stack en alternance, nous avons entamé ce processus ensemble. Ce fut également l'occasion de faire le tri dans nos données, en supprimant les tables inutiles et en ne conservant que celles essentielles.

Travaillant habituellement sur des bases de données locales, c'était la première fois que je m'attaquais à une migration vers une base de données en ligne. Cela m'a un peu intimidé au début, mais grâce à l'accompagnement de Chady, nous avons rapidement réussi à nous connecter à la base de données. C'est à ce moment-là que le véritable défi a commencé : la migration. Je n'avais que peu d'espoir que tout fonctionne du premier coup, et effectivement, nous avons dû nous y reprendre à plusieurs reprises avant de réussir à lancer les migrations automatiques fournies par Symfony.

Comme je m'y attendais, cela n'a pas marché dès le départ. Un problème survenu lors des migrations précédentes avait cassé le système de migration depuis plusieurs mois. Nous avons tenté à plusieurs reprises de résoudre ce problème avec Dan, mais en vain. Heureusement, Symfony propose une commande très utile qui permet de forcer la base de données à se conformer aux classes définies grâce à l'utilisation de Doctrine. Cette commande nous a fait gagner un temps précieux.

Mon premier réflexe a été d'essayer de régler les problèmes de migration, mais après plusieurs heures infructueuses, j'ai dû faire un choix. Devais-je continuer à m'acharner sur ces

---

<sup>22</sup> Chef du groupe chargé du développement en symfony avec moi

<sup>23</sup> <https://www.youtube.com/watch?v=lfmg-EJ8gm4>

problèmes, sur lesquels Dan et moi avons déjà bloqué pendant plusieurs jours quelques mois plus tôt, ou devais-je avancer sur le projet en utilisant cette solution rapide mais temporaire ? Ce choix, bien que pratique, ne résolvait pas le problème à sa source. Le principe des migrations et des fichiers de migration est de pouvoir disposer de sauvegardes (back-ups), très utiles en cas de problème avec la base de données. N'ayant pas réussi à faire fonctionner ce système correctement, la solution la plus simple était de forcer la base de données à s'aligner sur les classes via Doctrine. Toutefois, cela implique qu'il faut désormais prévoir des back-ups manuellement.

Une fois la migration terminée avec succès, nous avons été confrontés à un autre problème, matérialisé par cette erreur (voir ci-dessous) :

```
PDOException > Exception > SyntaxErrorException HTTP/1.1
An exception occurred while executing a query: SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'read, description, wallet_address, user_id, production_site_id) VALUES ('ROLE...' at line 1
```

Les erreurs SQL peuvent être résolues rapidement ou devenir des casse-têtes de plusieurs jours sans réelle avancée. Mon premier réflexe face à ce genre de situation est de copier-coller l'erreur sur Internet pour voir si une solution existe déjà. Sachant que cette page fonctionnait avant la migration, il ne s'agissait probablement pas d'une requête SQL mal écrite, un problème généralement facile à résoudre. Cependant, à seulement quelques heures de la présentation, cette erreur imprévue représentait une véritable catastrophe, d'autant que je n'avais aucune idée de sa cause exacte.

En parcourant divers forums, j'ai découvert qu'une solution potentielle pourrait consister à changer la version de la base de données. Par exemple, il était suggéré de passer de MariaDB <sup>24</sup>10.11 à une version antérieure, comme 10.5, pour résoudre ce type de problème. Cependant, cette solution semblait plus adaptée aux serveurs NextCloud<sup>25</sup>, et nous n'étions pas certains de son efficacité avec notre hébergement OVH. Après en avoir discuté avec notre sponsor, il nous expliqua que nous ne pouvions pas simplement changer de version de la base de données. La solution la plus simple aurait été de migrer vers une autre base de données, mais cela aurait pris trop de temps et ce n'étais pas sûr que cela résoudrait le problème.

Continuant mes recherches, j'ai sollicité l'aide de ChatGPT, mais sans succès. J'ai ensuite décidé de chercher des solutions spécifiques à Symfony en associant l'erreur à ce framework. J'ai alors trouvé un forum où une personne décrivait un problème similaire : une migration avec très peu de tables qui posait des soucis. En lisant le code et la solution proposée, j'ai appris que MariaDB réserve certains mots-clés pour son propre usage, et que ces mots ne doivent pas être utilisés comme noms de colonnes ou d'attributs dans les tables. Par exemple, il est très déconseillé d'utiliser des noms tels que "int" ou "integer" pour des variables en programmation, car ils sont réservés à la syntaxe du langage et peuvent causer des erreurs de compilation et autre.

---

<sup>24</sup> Un type de base de donnée

<sup>25</sup> Logiciel pour les données

Dans notre cas, SQLite n'avait aucun problème avec ces noms de colonnes, mais MariaDB, plus strict, n'acceptait pas qu'une colonne soit nommée "read". Après avoir identifié ce problème, j'ai simplement modifié toutes les requêtes SQL concernées et renommé toutes les colonnes et attributs "read" par un autre nom. Ce petit détail était en quelque sorte prévisible, mais il n'était pas évident de l'anticiper complètement. Il est compréhensible de ne pas utiliser des mots tels que "SELECT" ou "INSERT" pour des noms de colonnes, car ils sont essentiels aux commandes SQL. Cependant, le fait que SQLite accepte "read" tandis que MariaDB l'interdit suggère que MariaDB utilise probablement ce mot dans un autre contexte, ce qui explique pourquoi il est préférable de l'éviter pour ne pas créer de potentielles erreurs dans le code.

## Les objectifs ratés / non-complétés

### Modify resource admin

Comme l'indique le nom, cette mission consistait à permettre la modification d'une ressource directement via le tableau de bord administrateur. Cette fonctionnalité était déjà implémentée en Symfony, où il est relativement simple de modifier une ressource dans une base de données. En effet, pour changer un champ dans une base de données, il suffit de faire une requête SQL : on sélectionne la table appropriée, on filtre avec le bon ID pour trouver la ligne correspondante, puis on modifie le champ souhaité, comme par exemple le propriétaire de la ressource. Avec les outils fournis par Symfony et SQLite, cette opération est simple et rapide.

Cependant, implémenter cette fonctionnalité du côté de la blockchain était beaucoup plus compliqué. Les fonctionnalités de l'API blockchain étaient limitées, et intégrer cette fonctionnalité aurait nécessité une restructuration complète et des créations de plusieurs pages, ce qui aurait demandé au moins une journée de développement supplémentaire, une contrainte de temps que nous ne pouvions plus nous permettre.

Dans Symfony, la simplicité et les outils mise à disposition d'une base de données rend la modification des informations simple. En revanche, sur la blockchain, chaque modification aurait impliqué un input. Par exemple, si nous prenons le template<sup>26</sup> 'MetaDataTemplateMeat' (voir l'image de droite mentionnée précédemment), il aurait fallu créer une page dédiée avec au moins 12 input dont certains complexes comme les coordonnées GPS. Comment s'assurer que les coordonnées GPS sont valides ? Que faire si l'administrateur ajoute un chiffre en trop ? Devrions-nous permettre à l'administrateur de modifier des données calculées automatiquement, comme le temps de trajet (travelTime<sup>27</sup>) ? Ces questions ne représentent qu'une partie des problèmes auxquels j'avais envisagée lors de la potentiel refonte de cette fonctionnalité.

Face à ces défis, il m'a semblé évident qu'il n'était pas réaliste de développer cette fonctionnalité de manière complète sans sacrifier la qualité d'autres parties du projet. J'ai donc consulté les Masters pour leur expliquer la situation. Ils ont compris que même si cette fonctionnalité était importante, elle n'était pas absolument nécessaire dans le cadre du projet et qu'il

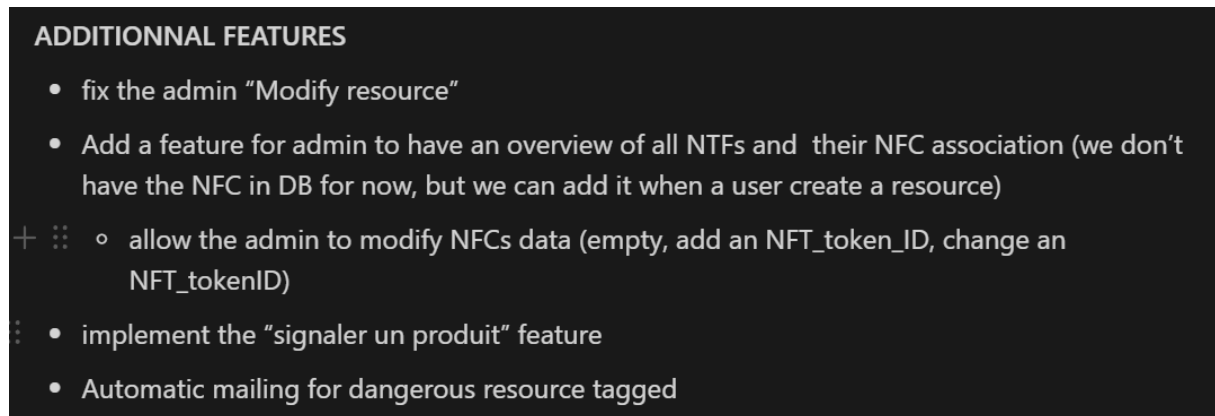
---

<sup>26</sup> Une structure type

<sup>27</sup> Calculé en faisant transportDate2 – transportDate1

valait mieux éviter de bâcler une partie du travail. Nous avons donc décidé de placer cette fonctionnalité dans la catégorie des 'FEATURES ADDITIONNELLES', à développer si le temps le permettait.

## Additional Features



Comme vous pouvez le voir ci-dessus, les "features additionnelles<sup>28</sup>" sont des fonctionnalités qui n'ont pas été spécifiquement demandées par les étudiants en Master 2, mais que nous avons envisagé d'implémenter. Parmi elles, il y avait l'implémentation de la fonctionnalité de signalement d'un produit. Cette fonctionnalité, initialement développée en Symfony, n'a pas pu être réintégrée dans la nouvelle version utilisant la blockchain.

La fonctionnalité de signalement permettait de signaler un produit dangereux via une page web. Idéalement, j'aurais souhaité que le signalement ne soit possible qu'après avoir scanné un tag, afin de garantir que seul le propriétaire légitime puisse signaler un produit qu'il possède. Cela aurait rendu le processus de signalement plus légitime et fiable en cas de besoin. Nous avons également déjà une section dédiée aux signalements de ressources dans le tableau de bord administrateur, mais cette fonctionnalité, également implémentée en Symfony, aurait nécessité beaucoup plus de temps pour être redéveloppée avec les moyens plus limités fournis par l'API blockchain.

Une solution de facilité aurait été de récupérer la ressource concernée et de simplement changer le paramètre 'isContaminated' à TRUE. Cependant, après avoir utilisé l'API à plusieurs reprises, je suis presque certain que cela aurait posé des problèmes, surtout pour les produits déjà utilisés. En théorie, cette fonctionnalité dépendrait beaucoup du développement côté blockchain. (La suite est théorique et s'appuie sur ce que j'ai appris pendant mes 2 jours de documentation blockchain) Nous utilisons une fonctionnalité appelée 'burn' pour indiquer qu'un NFT n'est plus possédé par personne et qu'il est "détruit". En réalité, cela revient à changer le propriétaire du NFT pour un propriétaire spécial avec un identifiant comme 00x0000...000. Cependant, obtenir des droits d'accès pour modifier des ressources appartenant à ce propriétaire spécial est très compliqué. J'ai souvent demandé ces droits, mais les étudiants en Master, et surtout Étienne, ne pouvaient ou ne

---

<sup>28</sup> Tous ce qui n'était pas possible de faire/ pas le temps



voulaient pas les donner. Je suppose donc qu'il s'agit d'un point délicat dans la blockchain qu'il vaut mieux éviter de toucher.

Pour résumer, bien que cette fonctionnalité puisse sembler simple à première vue, elle s'est révélée beaucoup plus complexe que prévu. Étant donné la complexité de la tâche, il aurait probablement été nécessaire de demander de l'aide aux Masters pour ajouter cette nouvelle fonctionnalité, ce qui aurait prolongé le temps de développement, une option qui n'était plus viable à ce stade du projet.

Enfin, concernant le "mailing automatique", cette fonctionnalité était directement liée à la précédente. Après avoir validé un produit comme dangereux via le tableau de bord administrateur, tous les utilisateurs ayant cherché un produit dont l'un des ingrédients provenait de la ressource contaminée auraient dû recevoir un mail. Cette fonctionnalité avait été initialement tentée par les étudiants en Licence 3 sur le projet. Cependant, après que Dan et moi-même avons cherché une solution, nous avons découvert que Google et Gmail avaient récemment désactivé cette fonctionnalité, ce qui nous a beaucoup gênés car c'était l'une des méthodes les plus couramment utilisées dans tous les exemples que nous avons trouvés. Nous avons donc décidé de mettre en attente le développement de cette partie du projet. Finalement, n'ayant pas développé la fonctionnalité de signalement de ressources, il était naturel de ne pas développer non plus la fonctionnalité de mailing.

## Bilan et expérience acquise

Malgré les trois semaines passées sur ce projet, j'ai l'impression que le temps a filé extrêmement vite. Avec un peu plus de recul, je regrette presque de ne pas avoir eu davantage de temps pour finaliser certaines fonctionnalités. Au début, je pensais que la mission serait simple et rapidement accomplie, mais après une semaine, des doutes ont commencé à s'installer. Ce qui devait prendre une semaine a finalement nécessité deux semaines sur les trois. L'implémentation des appels API et leur structuration a été un moment déterminant. Si je n'avais pas bien structuré le projet dès le départ, j'aurais perdu un temps précieux. Même si je n'ai pas pu appliquer une structuration en POO avec héritage comme prévu, cela m'a tout de même inspiré pour certaines parties. Ce n'est pas parce qu'on n'utilise pas un concept appris qu'il est inutile, parfois il nous aide à imaginer une autre structuration. Et cette structuration s'est révélée efficace pour terminer le projet dans les délais.

Ce que je regrette le plus dans ce projet, c'est peut-être mon entêtement sur certains points. Il y avait souvent des désaccords, surtout avec Quentin, avec qui j'ai probablement eu le plus de frictions. Nous n'étions pas d'accord sur certaines implémentations. Je cherchais la facilité, me disant : "Le programme que je propose produit le même résultat que ce que tu demandes," mais il n'acceptait pas ma méthode et m'imposait d'autres approches. Cela a conduit à plusieurs mini-disputes. J'ai fini par comprendre que c'était une demande spécifique des Master 2, qui souhaitaient un déroulement précis et non une autre approche. Je pense qu'à certains moments, j'ai laissé mon ego prendre le dessus, pensant souvent : "Je sais ce que je fais," car ayant développé tout le back-end du projet avec Dan, je connaissais bien le code et les avantages de Symfony. Alors, quand

Quentin me disait : "Je veux que ce soit fait de cette manière et pas autrement," j'avais du mal à me contenir, sachant qu'avec mes connaissances et les outils de Symfony, la méthode demandée serait plus longue à développer. Mais au fil du temps, j'ai décidé de me plier entièrement à ses demandes, ce qui a réduit les tensions lors des PR sur GitHub. Finalement, à la fin du projet, nous nous sommes tous retrouvés pour partager un dîner avec les M1 et moi-même. Nous avons pu discuter de tout ce qui s'était passé durant le projet, ce qui nous a beaucoup rapprochés. Cela m'a permis de mieux comprendre Quentin et certaines de ses décisions.

Enfin, le point le plus important de mon point de vue : je savais que j'avais une certaine facilité pour le développement et que j'adorais ça, mais j'ai toujours été un peu fainéant. Il m'arrivait de travailler plus de 10 heures par jour, mais seulement sur de courtes périodes. Voyant que je n'allais pas finir le projet à temps, je m'y suis personnellement investi à fond, d'où les 70 heures par semaine et certaines nuits où je me suis couché à 6h du matin. Me prouver que j'étais capable et que dans les moments critiques, on pouvait compter sur moi, est ma plus grande fierté. J'ai mal vécu la première semaine où je n'avançais pas du tout, sans réussir à compléter les exigences qui m'étaient données. Mais grâce à ma persévérance et mon acharnement, j'ai pu prouver ma capacité de travail et d'investissement aux Master, et plus important encore, à moi-même.

## Références

CoinHouse, Qu'est-ce qu'un smart contract ?

<https://www.coinhouse.com/fr/academie/ethereum/smart-contract/> , consulté en aout 2024

LeMagIT, 25 % des projets en France ne respectent pas leurs délais

<https://www.lemagit.fr/actualites/2240196586/25-des-projets-en-France-ne-respectent-pas-leurs-delais>, consulté en aout 2024

Clubic, Qu'est-ce qu'un Raspberry Pi ? Introduction au nano-ordinateur

<https://www.clubic.com/raspberry-pi/article-849782-1-raspberry-pi-introduction-nano-ordinateur.html>, consulté en aout 2024

It-Connect, Autoriser le SSH via Iptables

<https://www.it-connect.fr/autoriser-le-ssh-via-iptables/> , consulté en aout 2024

SpeedGuid, Port 20 Details

<https://www.speedguide.net/port.php?port=20>, consulté en aout 2024

Ionos, Ports TCP/UDP : liste des ports les plus importants

<https://www.ionos.fr/digitalguide/serveur/know-how/ports-tcpet-udp/>, consulté en aout 2024

Blog.netwrix, Liste des vulnérabilités liées aux ports ouverts

<https://blog.netwrix.fr/2023/07/20/liste-des-vulnerabilites-liees-aux-ports-ouverts/>, consulté en aout 2024

Specopssoft, Les ports ouverts et leurs vulnérabilités

<https://specopssoft.com/fr/blog/les-ports-ouverts-et-leurs-vulnerabilites/>, consulté en aout 2024

FreeDesktop,

<https://www.freedesktop.org/software/systemd/man/latest/systemd-journald.service.html>, consulté en aout 2024

Repost, Comment faire pour corriger les erreurs CORS liées à API Gateway ?

<https://repost.aws/fr/knowledge-center/api-gateway-cors-errors>, consulté en aout 2024

---

<sup>i</sup> <https://www.coinhouse.com/fr/academie/ethereum/smart-contract/>