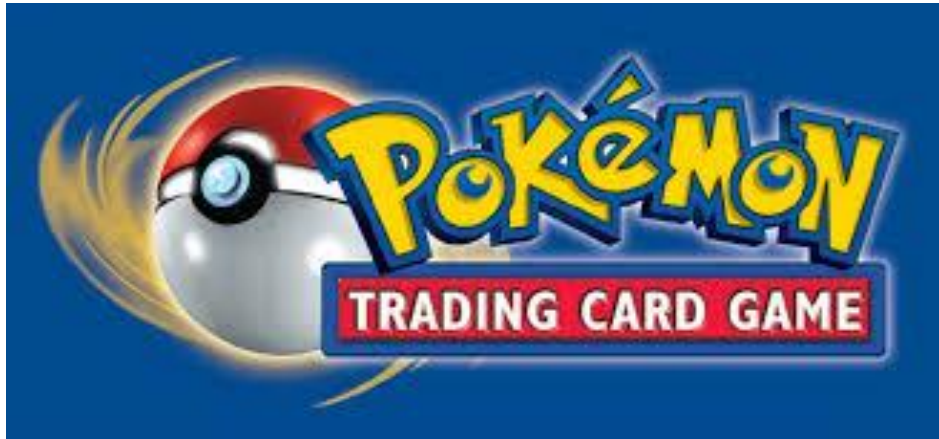


Rapport – Projet Final : Pokémon Base Set NFT



Par Camille Guillard

Email : camille-guillard@outlook.fr

GitHub : <https://github.com/camille-guillard?tab=repositories>

Linkedin : <https://www.linkedin.com/in/camille-guillard-b1071b1b8/>

Formation Acadee : Exploiter la blockchain dans le développement d'applications Web3

Remerciements :

Lorcann RAUZDUEL (Formateur)

Samuel Dumas (Fondateur d'Acadee)

Table des matières

I - Introduction	3
II – Partie fonctionnelle	3
1 - Carte à jouer et à collectionner.....	3
2 - Bien positionnel	4
3 - Reconnaissance sociale et sentiment d'appartenance à une communauté	4
4 - Passage au web2.....	5
5 - Apport de la blockchain	6
6 - Point de vue business	6
III – Partie technique	7
1 - Présentation du projet	7
2 - Structure du projet	7
3 - Contrat	8
4 - Tests Unitaires, Sécurité, optimisation	9
5 - Utiliser l'algorithme Merkle Tree pour la Whitelist.....	11
6 - Tirage des cartes et gestion de l'aléatoire	12
7 - Déploiement sur Sepolia.....	12
8 - Interface utilisateur	14
IV - Conclusion.....	17

I - Introduction

Mon projet NFT est basé sur la première série de cartes à jouer et à collectionner Pokémon sortie en 1996 au Japon et 1999 en Amérique du Nord et en Europe, le « Set de Base ». Elle a été éditée par Wizards of the Coast, et contient un total de 102 cartes. Si elle n'est pas une extension à proprement parler (une extension venant étendre le set de base), elle est généralement considérée comme telle par simplicité.

Les cartes étaient vendues par paquet de 11 cartes (un booster) ou dans un carton de 36 paquets de carte, soit 396 cartes (un display). Chaque carte avait un type défini (pokémon, entraîneur ou énergie) et un niveau de rareté (du moins rare au plus rare : commune, peu commune, rare, holographique). Chaque paquet de carte, contenait obligatoirement 2 cartes énergie communes, 5 cartes communes (pokémon ou entraîneur), 3 cartes peu communes (pokémon ou entraîneur) ; quant à la dernière, il y avait 1 chance sur 3 de tirer une carte holographique (pokémon) ou bien 2 chances sur 3 de tirer une carte rare (pokémon ou entraîneur).

Liste complète des 102 cartes : <https://www.pokecardex.com/series/BS>

Le projet sera déployé sur la blockchain Testnet de Ethereum : Sepolia (chainId 11155111)

II – Partie fonctionnelle

1 - Carte à jouer et à collectionner

La première fonctionnalité évidente est au niveau des jeux de cartes à jouer et à collectionner.

Pour jouer au jeu, il faut d'abord que 2 joueurs composent un deck de 60 cartes en achetant des cartes.

Voici la première version des règles du jeu datant de 1999 :

<https://www.docdroid.net/h266/nintendo-power-1999-pokemon-trading-card-game-pdf>

Le joueur souhaitant s'améliorer, ou plus, de jouer dans un système compétitif devra améliorer son deck en achetant d'autres packs, sur le marché secondaire, où en échangeant ses cartes avec d'autres joueurs.

L'utilisateur souhaitant récupérer les 102 cartes devra également acheter des nouveaux boosters ou/et en acheter ou en échanger sur le marché secondaire.

Attention, le phénomène de collectionnisme qui émerge de la passion peut rapidement tourner à l'obsession. Le collectionneur prend plaisir à la complétion lors de ce voyage en ajoutant chaque nouvelle pièce à sa collection mais perd rapidement ce plaisir une fois celle-ci finie contrairement à ce qui était attendu, recommençant ainsi une nouvelle collection. C'est pourquoi il peut être dangereux pour son portefeuille s'il perd le contrôle de ses achats.

2 - Bien positionnel

Le second aspect que je souhaite mettre en valeur est qu'une carte rare peut être un bien positionnel.

Un bien positionnel est un objet qui détermine le niveau social d'une personne par son statut exclusif (généralement déterminé par l'argent ou la position sociale). Un objet de luxe ne l'est pas forcément, car il faut qu'il soit visible ou connu. On peut notamment citer comme exemple les montres, les voitures, les récompenses artistiques ou sportives, les œuvres d'art, les lieux d'habitation, etc.

Le bien positionnel a également ceci de particulier : il n'est pas éternel. S'il devient trop courant, il cesse d'être un différenciateur suffisant. Typiquement, une voiture simple n'est plus positionnelle, pas plus qu'un téléphone.

Un point important à noter est que cet objet sert à attiser généralement l'envie des autres. Il permet de montrer aux autres notre réussite sans avoir à le dire directement.

Dans mon exemple, il s'agirait par exemple de montrer au monde entier sa nouvelle carte « Dracaufeu holographique » (carte très appréciée par la communauté de carte Pokémon).

3 - Reconnaissance sociale et sentiment d'appartenance à une communauté

Beaucoup de personnes ont une aversion négative envers les NFT ou les meme coins car la valeur intrinsèque est proche de zéro. Mais pour quelles raisons des NFT de singes ou des cryptomonnaies à tête de chien peuvent valoir un aussi gros montant ?

Il y a certes un phénomène de spéculation, mais qui peut être présent sur n'importe quel actif, je vais donc me concentrer simplement sur la base de mouvement.

Je pense que la réponse se trouve dans le sentiment d'appartenance à un groupe qui a été très important pour l'évolution de l'humanité. À l'époque primitive, faire partie du groupe augmentait vos chances de survie. Le sentiment d'exclusion peut également être désagréable en raison de la façon dont il est traduit dans le cerveau. La recherche

montre que le rejet social peut être interprété par les mêmes régions du cerveau que celles responsables du traitement de la douleur physique. À l'inverse, être reconnu socialement offre un sentiment de bien-être, car nous avons gardé cet aspect primitif qui nous apporte un sentiment de sécurité qui a contribué à notre survie.

Ce phénomène est proche du pourquoi de simple sac-à-main se vendent 3000€, où des montres à 50000€, qui ont des valeurs démesurées pour leurs simple utilisations intrinsèques et qui se révèle être simplement un bien positionnel, qui permet d'être vu et validé par la communauté. On pouvait entendre il y a 10 ans que LVMH était surévalué due à un PER trop élevé (Price Earning Ratio ou ratio cours sur bénéfices) ce qui ne l'a pas empêché par la suite de battre à maintes reprises son plus haut historique, ou bien que le DogeCoin ne valait rien et qu'il serait bientôt à 0, ce qui ne l'a pas empêché de survivre à 2 bearmarkets et d'être toujours dans le top 10 de capitalisation.

Il peut parfois être rageant pour un développeur ou un investisseur de constater qu'une crypto-monnaie à tête de chien, le Dogecoin capitalisé à plus de 54 milliards de dollars et fondé à partir d'un fork du Litecoin dans le but d'en faire une simple blague, est valorisé 23 fois plus que le token AAVE en termes de capitalisation (2.2B), un top protocole de finance décentralisé et 2 fois plus que toute la liquidité investie dans ce protocole (27B) à l'heure où j'écris ces lignes. Mais il faut savoir prendre du recul pour nous comprendre et nous accepter en tant qu'être humain.

Le marché n'est pas toujours rationnel, ainsi certains acteurs peuvent faire un achat ou une vente sous le coup de l'émotion, tomber amoureux de leurs cryptoactifs, et finalement finir par le défendre sur internet comme s'il s'agissait d'une équipe de foot. Et même si le marché n'est pas forcément rationnel, il faut admettre qu'il a toujours raison.

4 - Passage au web2

Depuis 25 ans, le jeu de cartes au format papier est toujours d'actualité avec de nombreuses séries de cartes sont sorties depuis et est toujours aussi populaire auprès des fans.

Mais il est également possible grâce à la démocratisation d'internet de pouvoir jouer en ligne avec des cartes virtuelles dans le Pokémon TCG (Trading Card Gamme) ou JCC en français (Jeu de Cartes à Collectionner) : <https://tcg.pokemon.com/fr-fr/>

Cependant, la centralisation entraine un manque de transparence sur les données de jeu (comme par exemple la supply maximum) et limite l'utilisation des cartes qu'au logiciel de la TCG et avec les règles limitées de l'application (l'éditeur peut refuser la vente ou l'échange par exemple s'il le souhaite). Imaginons que l'éditeur souhaite rajouter une carte qui était rare à la base, elle aurait beaucoup moins de valeur à l'unité. Ou bien que l'éditeur sorte une nouvelle version du jeu chaque année (et où il faut tout

racheter) puis arrête subitement les serveurs des anciens jeux au bout de quelques années compromettant ainsi l'accès aux cartes par leurs propriétaires comme c'est le cas pour FIFA Ultimate Team.

5 - Apport de la blockchain

Compte tenu des éléments apportés, la blockchain permet de ramener une relation de confiance en passant par un contrat entre l'éditeur et l'utilisateur dû à sa transparence (une fois le contrat vérifié) et son immuabilité (impossible d'augmenter le supply.

De plus, grâce à la décentralisation, l'utilisateur peut interagir avec n'importe quel autre site web3 en connectant son portefeuille et utiliser son NFT si l'application s'il propose des fonctionnalités. Il a également la possibilité de vendre son NFT sur un site d'échange, comme OpenSea par exemple, s'il respecte bien les normes ERC-721. L'utilisateur possède bien son NFT.

6 - Point de vue business

Comme le montre les points précédents, le projet est intéressant d'un point de vue business dans le domaine du divertissement. D'ailleurs, les équipes en marketing utilisent couramment les études en psychologie humaines afin de mieux vendre leurs produits.

De plus, les apports de la blockchain amènent de la transparence qui renforce la confiance entre l'éditeur et les utilisateurs, ce qui est très important puisque la valeur d'une carte dépend de sa rareté.

Dans un marché d'offres et de demandes, il ne suffit pas qu'un bien soit rare pour avoir de la valeur, mais également qu'il suscite une forte demande, ce qui est le cas pour la licence Pokémon qui a formé une grande communauté sur ces 25 dernières années.

Bien sûr dans les faits, mon NFT n'a aucune valeur puisque j'en suis l'éditeur, mais je suis prêt à parier qu'il aurait beaucoup de valeur s'il était édité en quantité limitée par « The Pokémon Company » qu'il y ait un rôle dans le jeu de carte en ligne ou pour la collection seulement.

III – Partie technique

1 - Présentation du projet

Période de prévente :

Pendant la période des ventes privées, les utilisateurs inscrits par le propriétaire du contrat lors du déploiement peuvent acheter 2 boosters au maximum. Après l'avoir acheté, les utilisateurs peuvent miner les 11 nft de chaque booster.

Période de ventes publiques :

Pendant la période des ventes publiques, tous les utilisateurs peuvent acheter des boosters et des displays. Il est possible d'acheter un maximum de 35 boosters à la fois et un maximum de 6 présentoirs à la fois. Un utilisateur peut stocker un maximum de 216 boosters non ouverts. L'utilisateur peut ensuite ouvrir ses boosters tant qu'il en a en stock.

Le contrat propose une offre maximum de 3 600 000 boosters.

Le prix d'un booster est fixé à 0.01 ether tandis que le prix d'un display est fixé à 0.3 ether (au prix de 30 boosters).

Pour la composition des cartes des boosters, le même algorithme que la version d'origine a été utilisé, soit : 2 cartes énergie communes, 5 cartes communes, 3 cartes peu communes, 1 chance sur 3 de tirer une carte holographique ou bien 2 chances sur 3 de tirer une carte rare.

2 - Structure du projet

Le projet contient un contrat codé avec solidity 0.8.28 et est déployé avec hardhat.

Il y a également un projet front réalisé en NextJs :

Voici une description des fichiers les plus importants du projet

`./config/registeredAddressForWhitelist.js` : liste des adresses de la whitelist

`./contracts/PokemonBaseSet.sol` : contrat du projet

`./front` : répertoire source du front réalisé en NextJS

`./ignition/PokemonBaseSetModule.js` : script de déploiement

`./ignition/herlper-config.js` : variables utilisées pour le déploiement

`./metadata` : metadata de chaque carte qui sera uploadé vers un cloud

./ scripts/merkleTree.js : script permettant de générer l'arbre de merkle
./test/ PokemonBaseSet.test.js : tests unitaires
hardhat.config.js : configuration hardhat pour le déploiement
./env : fichier de propriétés contenant les clés privé du wallet et les clés des API (non inclus sur le répertoire git)

3 - Contrat

Le contrat étant la norme ERC-721, token non fongible sur la blockchain Ethereum. Cette norme de tokens est utilisée pour la majorité des tokens non fongibles (NFT) déployés sur cette blockchain.

Initialisation du contrat :

string memory _nftURI : lien vers le cloud contenant metadonnées
address _claimFundAddress : adresse de la personne qui pourra réclamer les fonds
uint256 _preSalesStartTime : date de début de la période de la vente privée
uint256 _preSalesEndTime : date de fin de la période de la vente privée
uint256 _publicSalesStartTime : date de début de la période de la vente publique
bytes32 _merkleTreeRootHash : adresse racine de l'arbre de merkle
uint256 _subscriptionId : numéro de souscription VRF2.5
address _vrfCoordinator : l'adresse du contrat utilisé par VRF
bytes32 _keyHash : hash correspondant au prix maximum du gaz
uint32 _callbackGasLimit : limite de gaz pour l'exécution du callback VRF
uint16 _requestConfirmations : nombre de blocs de confirmation nécessaire

Variables :

- userBoosters : chaque adresse est mappé à la structure UserBoosters, contenant le nombre de booster non ouverts (numberOfUnopenedBoosters, et tout le temps inférieur à 216), l'index actuelle de la carte à ouvrir (currentRandomNumberIndex), l'identifiant de la dernière, requête vers VRF (requestId), un boolean pour vérifier si la tirage des nombres aléatoires et toujours en cours, et les résultats des tirages de nombre aléatoires sur VRF.
- holoCardIndex : tableau des index des cartes holographiques
- rareCardIndex : tableau des index des cartes rares
- uncommonCardIndex: tableau des index des cartes peu communes
- commonCardIndex : tableau des index des cartes communes
- commonEnergyCardIndex : tableau des index des cartes energies communes
- cardIds : chaque index de nft sera associé à un identifiant de carte (entre 0 et 101)

- nftURI : lien racine vers le cloud contenant les métadonnées

Fonctions principales :

- preSaleBuyBooster : L'utilisateur fournit la preuve qu'il est dans la whitelist et le nombre de booster souhaité. S'il en demande 1 ou 2 et qu'il est dans la whitelist, l'utilisateur pourra générer ses boosters.
- buyBooster : Tout utilisateur peut acheter entre 1 et 35 dès l'ouverture de la vente publique
- buyDisplay : Tout utilisateur peut acheter un display dès l'ouverture de la vente publique
- drawCards : Cette fonction interne sera appelé par les autres méthodes permettant d'acheter un booster. L'oracle VRF2.5 sera alors appelé dans le but de générer un nombre aléatoire pour chaque booster.
- fulfillRandomWords : Méthode de callback interne permettant de récupérer les nombres aléatoires et de les associer aux nouveaux boosters en stock.
- openBooster : L'utilisateur aura la possibilité d'ouvrir un booster s'il n'y a pas de tirage en cours et s'il possède au moins 1 booster. La méthode permet de minter les 11 cartes (en respectant l'algorithme de rareté) et en sélectionnant la carte grâce au nombre aléatoire généré pour le booster (et en ajoutant le numéro d'itération i, nombre déterministe non manipulable pour ne pas toujours sortir les mêmes cartes sur les mêmes types de rareté). Le numéro du NFT sera associé à l'index de la carte dans le mapping cardIds.
- tokenURI : permet de récupérer le lien vers les métadonnées d'un NFT à partir de son numéro (nftURI + numéro de la carte entre 0 et 101)

Certaines fonctions permettent de modifier des valeurs une fois le contrat déployé comme le prix, les dates, les paramètres VRF, le hash racine de la whitelist. Cependant, la supply maximum restera immuable.

4 - Tests Unitaires, Sécurité, optimisation

Lors de ma formation, j'ai été guidé par mon formateur pour produire un code de qualité pour chacun de mes projets.

Les tests unitaires ont été effectués dans le fichier ./test/ PokemonBaseSet.test.js , où chaque scénario a été testé, même en cas d'échec. Pour le tester, il faut compiler le contrat, déployer un node en local, et lancer les tests sur le réseau localhost.

En ligne de commande :

Compiler : npx hardhat compile

Créer un node en local : npx hardhat node

Lancer les tests unitaires : npx hardhat test --network localhost

Les types des variables ont été optimisés afin de réduire le coût en gaz.

L'accessibilité aux fonctions ont été correctement étudié afin d'empêcher une mauvaise utilisation du contrat.

Les différents types d'attaques les plus courantes ont été étudiés afin d'améliorer la sécurité (OWASP).

Optimisation :

À la fin du développement, j'ai installé « hardhat gas reporter » pour avoir une estimation du gaz consommé lors des tests. La fonction « openBooster » consommait beaucoup trop de gaz, c'est pour cette raison que j'ai dû faire 2 optimisations.

Solidity and Network Configuration					
Solidity: 0.8.28		Optim: true	Runs: 200	viaIR: false	Block: 30,000,000 gas
Methods					
Contracts / Methods	Min	Max	Avg	# calls	usd (avg)
PokemonBaseSetMock					
buyBooster	201,487	255,587	205,222	234	-
buyDisplay	204,122	258,222	250,493	7	-
drawCards	-	-	219,915	4	-
openBooster	1,587,013	1,609,002	1,603,173	53	-
preSaleBuyBooster	207,043	278,243	260,793	8	-

La première était d'utiliser ERC721A pour pouvoir minter les 11 cartes à la fois lors de l'ouverture d'un booster :

solc version: 0.8.28		optimizer enabled: true		Runs: 200	Block limit: 6718946 gas	
Methods						
Contract	Method	Min	Max	Avg	# calls	usd (avg)
PokemonBaseSetMock	buyBooster	203571	257671	207306	234	-
PokemonBaseSetMock	buyDisplay	206228	260328	251311	6	-
PokemonBaseSetMock	drawCards	-	-	219893	3	-
PokemonBaseSetMock	openBooster	369597	413397	380647	58	-
PokemonBaseSetMock	preSaleBuyBooster	207021	278221	254954	6	-

La seconde était de remplacer la recherche dans le tableau par un calcul sur les extrémités de la bordure des index :

Solc version: 0.8.28		Optimizer enabled: true		Runs: 200	Block limit: 6718946 gas	
Methods						
Contract	Method	Min	Max	Avg	# calls	usd (avg)
PokemonBaseSetMock	buyBooster	203571	257671	207306	234	-
PokemonBaseSetMock	buyDisplay	206228	260328	251311	6	-
PokemonBaseSetMock	drawCards	-	-	219893	3	-
PokemonBaseSetMock	openBooster	350791	395684	362026	58	-
PokemonBaseSetMock	preSaleBuyBooster	206993	278193	254926	6	-
PokemonBaseSetMock	setBaseURI	-	-	28687	2	-

5 - Utiliser l'algorithme Merkle Tree pour la Whitelist

En informatique et en cryptographie, un arbre de Merkle ou arbre de hachage est une structure de données contenant un résumé d'information d'un volume de données, généralement grand (comme un fichier). L'algorithme de l'arbre Merkle est couramment utilisé comme données cryptées pour vérifier si une adresse est incluse ou non dans une liste blanche.

Un arbre Merkle est un arbre dans lequel chaque nœud « feuille » est étiqueté avec le hachage cryptographique d'un bloc de données, et chaque nœud qui n'est pas une feuille (appelé branche, nœud interne ou inode) est étiqueté avec le hachage cryptographique des étiquettes de ses nœuds enfants. Un arbre de hachage permet une vérification efficace et sécurisée du contenu d'une grande structure de données. Un arbre de hachage est une généralisation d'une liste de hachage et d'une chaîne de hachage.

Génération de la whitelist

Les adresses incluses dans la whitelist doivent être placées dans le tableau du fichier `./config/registeredAddressForWhitelist.js`. Une fois fait, il faut exécuter le script suivant :

```
node scripts/merkleTree.js
```

```

MINGW64:/c/Users/camil/workspace/my-nft
camil@Camille MINGW64 ~/workspace/my-nft (main)
$ node scripts/merkleTree.js
Here is Root Hash: 0x12014c768bd10562acd224ac6fb749402c37722fab384a6aecc8f91aa7dc51cf

camil@Camille MINGW64 ~/workspace/my-nft (main)
$ |

```

Le hash racine sera retourné en réponse pour l'ajouter aux paramètres du contrat. De plus, un nouveau fichier nommé « `whiteList.json` » sera créé contenant la feuille et la preuve de toutes les adresses de la liste blanche. Pendant la période de prévente, l'utilisateur pourra prouver qu'il fait bien partie de la whitelist en présentant l'adresse preuve associé à son adresse publique. S'il est bien vérifié, cela veut dire qu'il fait bien

partie de la whitelist et qu'il pourra acheter les 2 boosters lors de la période de prévente.

6 - Tirage des cartes et gestion de l'aléatoire

Lors de la première implémentation du contrat, j'avais fait l'erreur de mettre des valeurs pouvant être manipulées par l'utilisateur comme par exemple le timestamp. Ce qui aurait pu lui donner la possibilité de prédire quand exécuter sa transaction afin de ne tirer que la carte holographique qu'il voulait. Mon formateur m'a donc proposé d'utiliser un oracle afin de tirer des nombres aléatoires hors de la blockchain.

En effet, l'aléatoire est très difficile à générer sur les blockchains. Chaque nœud de la blockchain doit parvenir à la même conclusion et former un consensus. Même si les nombres aléatoires sont polyvalents et utiles dans diverses applications blockchain, ils ne peuvent pas être générés de manière native dans les contrats intelligents. La solution à ce problème est Chainlink VRF, également connu sous le nom de Chainlink Verifiable Random Function.

Juste après l'achat d'un booster, une demande est envoyée à vrf chainlink pour récupérer des nombres aléatoires en dehors de la blockchain. La fonction callback récupère une liste de nombres aléatoires (un pour chaque booster) et ces données sont enregistrées. Une fois les numéros aléatoires disponibles, l'utilisateur peut désormais ouvrir les boosters pour découvrir ses cartes. Ce nombre aléatoire externe et la position de la carte sont utilisés pour sélectionner la carte qui sera tirée, variables déterministes, empêchant ainsi l'utilisateur de tricher en prédisant les cartes qui lui seront attribuées grâce à la manipulation des paramètres.

Documentation de démarrage avec Chainlink VRF V2.5 : <https://docs.chain.link/vrf/v2-5/getting-started>

7 - Déploiement sur Sepolia

Metadata :

Ajouter le dossier ./metadata sur le cloud et utiliser le lien dans le constructeur.

<https://gold-acute-python-408.mypinata.cloud/ipfs/QmPS91JPSF4qLxixrNFhAso4FjJciEaCe8Fefr5THd6miX/>

Script de déploiement :

```
npx hardhat ignition deploy ./ignition/modules/PokemonBaseSetModule.js --network sepolia
```

Script de vérification du contrat :

npx hardhat verify --network sepolia

```
0xABcBa62bab2662fa7a23996773dbCB75652d94FA "https://gold-acute-python-408.mypinata.cloud/ipfs/QmPS91JPSF4qLxixrNFhAso4FjJciEaCe8Fefr5THd6miX/" "0x907D577AF8386402e248AD3736593627cAFD1085" "1727654400" "1728259199" "1728259200" "0x12014c768bd10562acd224ac6fb749402c37722fab384a6aecc8f91aa7dc51cf" "113976695910214076856129855253409653665408242942060079584559527527298385366102" "0x9DdfaCa8183c41ad55329BdeeD9F6A8d53168B1B" "0x787d74caea10b2b357790d5b5247c2f63d1d91572a9846f780606e4d953677ae" "900000" "3"
```


Contract vérifié

<https://sepolia.etherscan.io/address/0xD14d0A4Cc4BD6af4686c03Bc6Fe46782e8BFbb77#code>


Chainlink VRF2.5 :

Ajouter l'adresse du contrat à la souscription Chainlink :<https://vrf.chain.link/#/side-drawer/subscription/113976695910214076856129855253409653665408242942060079584559527527298385366102>

Subscription ...6102 ▾ | Overview History

Status	Network	ID ⓘ	Admin ⓘ
● Active	⚡ Ethereum Sepolia	113976...6102 ⓘ	 0x907d...1085 ⓘ
Consumers	Fulfillments ⓘ	Version	Balance ⓘ
1	33	2.5	620.937419438443 LINK, 0.01 ETH

Consumers

Address	Added ⓘ	Last fulfillment ⬇	Total spent ⓘ	Actions
 0xabcb...94fa ⓘ	November 29, 2024 at 15:47 UTC	November 29, 2024 at 20:11 UTC	0.43864328809820013 LINK, 0 ETH	⋮

Adresse publique du portefeuille utilisé :

0x907D577AF8386402e248AD3736593627cAFD1085

Adresse du contrat sur Sepolia (chainId : 11155111) :

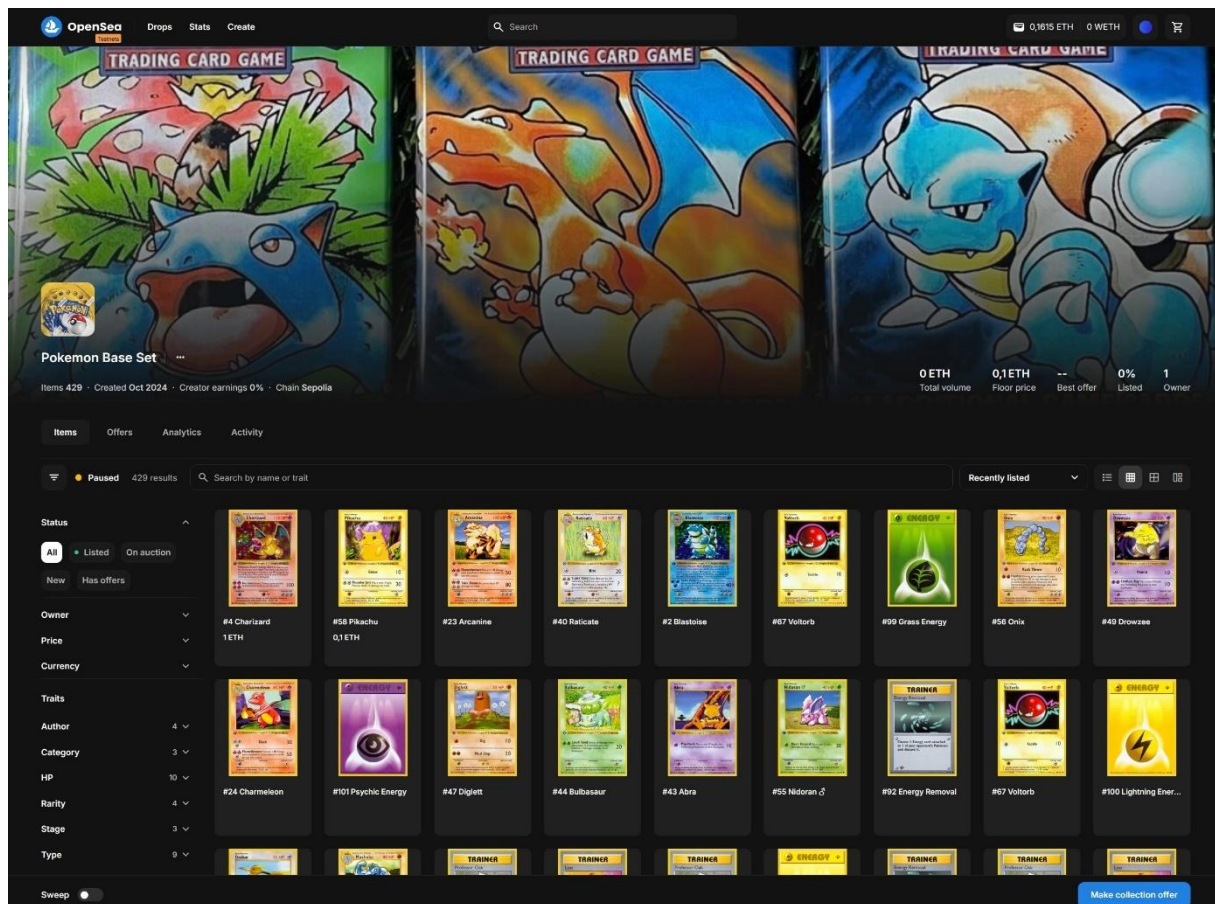
0xABcBa62bab2662fa7a23996773dbCB75652d94FA

Chainlink VRF2.5 subscription

<https://vrf.chain.link/#/side-drawer/subscription/113976695910214076856129855253409653665408242942060079584559527527298385366102>

OpenSea

[https://testnets.opensea.io/collection/pokemon-base-set?search\[sortAscending\]=false&search\[sortBy\]=LAST_TRANSFER_DATE](https://testnets.opensea.io/collection/pokemon-base-set?search[sortAscending]=false&search[sortBy]=LAST_TRANSFER_DATE)



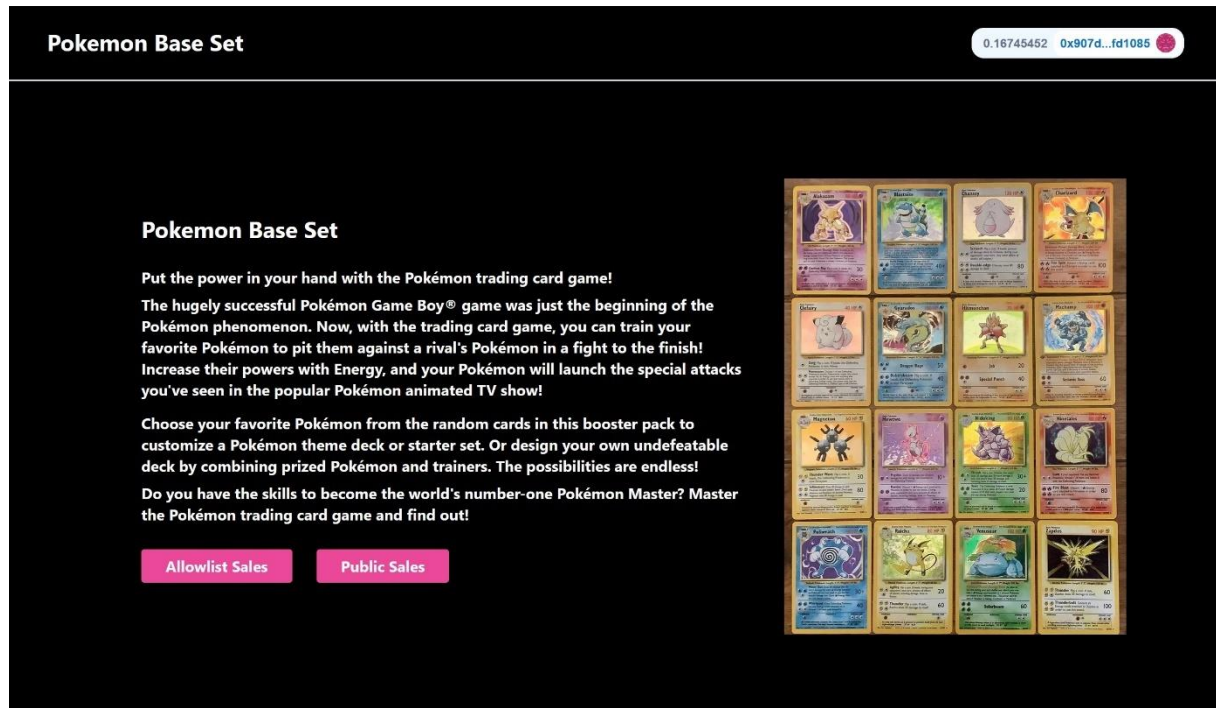
8 - Interface utilisateur

Un fois le contrat compilé, il est possible de récupérer l'abi à la ligne 5 du le fichier `./artifacts/contracts/PokemonBaseSet.sol/PokemonBaseSet.json`. L'abi est la signature sur contrat qui nous permettra d'interagir avec notre contrat depuis notre grâce à la librairie moralis, elle sera ajouté dans un fichier de constant `./front/constants/`

pokemon-base-set-abi.json, tandis que l'adresse du contrat déployé sur sepolia se trouvera dans le fichier ./front/constants/ pokemon-base-set-address.json.

Le dossier ./front/pages contiennent les pages presale et mint, qui permettent et utilisent leurs composants respectifs. La page presale permet de minter ses boosters lors de la période de prévente et la page mint pendant la période de vente publique.

Sur la page /home se trouve une présentation de la collection et 2 boutons permettant de naviguer vers la page /presale et /mint.



Sur la page des composants, j'ai utilisé la méthode useWeb3Contract de moralis pour interagir avec les différentes fonctions de l'abi.

Les informations affichées à l'écran sont rafraîchies grâce aux hooks de React.

Le formulaire s'affiche seulement s'il est dans la bonne période de vente.


L'utilisateur peut acheter des boosters pour voir son compteur de boosters en stock augmenter. Un fois générés, il peut commencer à les ouvrir et à minter ses NFT.

Page de mint :

Pokemon Base Set


0.16187944 0x907d...fd1085

Public Sale Page
Total Supply : 429 / 3600000



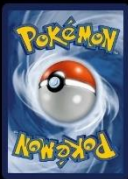
Buy booster(s) :

Buy a booster



1 display : 36 boosters !

Buy a display




Number of unopened boosters : 10

Open a booster

Achat de 3 boosters :


Pokemon Base Set

Public Sale Page
Total Supply : 396 / 3600000

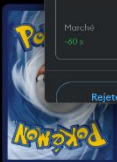


Buy booster(s) :

Buy a booster



Buy a display



Number of unopened boosters : 8

Open a booster

http://localhost:3000
0xD14d0...Fbb77 INTERACTION AVEC UN CONTRAT
0.03 SepoliaETH

DÉTAILS HEX

Changements estimés
Vous envoyez -0.03 SepoliaETH 74,64 BUS

Frais estimés
0.00024653 0.00024653 SepoliaETH
Marché -60 s Frais maximaux : 0.00028367 SepoliaETH


Rejeter Confirmer

Ouverture d'un booster :

Pokemon Base Set


Public Sale Page

Total Supply : 396 / 3600000

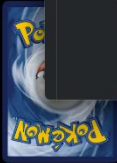


Buy booster(s) :

Buy a booster



Buy a display



Number of unopened boosters : 8

http://localhost:3000

0xD14d0...Fbb77 INTERACTION AVEC UN CONTRAT

DÉTAILS HEX

Changements estimés ⓘ

Vous recevez

+ #397

0xD14d0...Fbb77

+ #398

0xD14d0...Fbb77

+ #399

0xD14d0...Fbb77

+ #400

0xD14d0...Fbb77

+ #401

0xD14d0...Fbb77

+ #402

0xD14d0...Fbb77

IV - Conclusion

J'ai choisi de vous présenter ce projet car c'est sur celui-ci où j'ai pu progresser le plus sur pleins d'aspects (solidity, hardhat, tests, sécurité, algorithmique, oracle, interface utilisateur, ...)

Mais c'est aussi le projet où j'ai été le plus inspiré, et qui m'a ramené en enfance, lors de la fin des années 90 et qui parlera aux gens de ma génération.

Merci d'avoir pris le temps d'étudier mon rapport.

Sources :

<https://www.pokecardex.com/series/BS>

[https://bulbapedia.bulbagarden.net/wiki/Base_Set_\(TCG\)](https://bulbapedia.bulbagarden.net/wiki/Base_Set_(TCG))

<https://www.psychologies.com/Moi/Se-connaître/Personnalité/Articles-et-Dossiers/Les-collectionneurs-sont-ils-nevroses>

<https://mentorshow.com/blog/sentiment-mis-a-l-ecart>

<https://mailchimp.com/fr/resources/marketing-psychology-principles/>

<https://ethereum.org/fr/developers/docs/standards/tokens/erc-721/>

<https://hardhat.org/docs>

https://fr.wikipedia.org/wiki/Arbre_de_Merkle

<https://medium.com/@ItsCuzzo/using-merkle-trees-for-nft-whitelists-523b58ada3f9>

<https://dev.to/muratcanyuksel/checking-whitelisted-addresses-on-a-solidity-smart-contract-using-merkle-tree-proofs-3odm>

<https://medium.com/@dogukanakkaya/how-to-mock-chainlink-vrf-coordinator-v2-and-aggregator-v3-with-truffle-0-8-0-24353b96858e>

https://github.com/smartcontractkit/chainlink/blob/develop/contracts/src/v0.8/vrf/mocks/VRFCoordinatorV2_5Mock.sol

https://securing.github.io/SCSVS/SCSVS_v1.1.pdf

<https://owasp.org/www-project-smart-contract-top-10/>

<https://lorcannrauzduel.substack.com/p/les-10-principales-faibles-des-smart>