Vývoj Web3 NFT herní aplikace

Bc. David Rábel

Diplomová práce
2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. David Rábel**
Osobní číslo:	**A22474**
Studijní program:	**N0613A140022 Informační technologie**
Specializace:	**Softwarové inženýrství**
Forma studia:	**Kombinovaná**
Téma práce:	**Vývoj Web3 NFT herní aplikace**
Téma práce anglicky:	**Development of a Web3 NFT Game Application**

Zásady pro vypracování

1. Vypracujte literární rešerši na zadané téma.
2. Vytvořte NFT kolekci ve vhodném standardu.
3. Navrhněte a zrealizujte databázi pro NFT kolekce.
4. Vytvořte webovou minting a informační aplikaci.
5. Věnujte pozornost zabezpečení webové aplikace.
6. Implementujte backendovou část pro herní aplikaci ve formě Unity Asset Package.
7. Vhodně popište postup a implementaci řešení.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. VAN HIJFTE, Stijn. Blockchain Platforms: A Look at the Underbelly of Distributed Platforms. Howest Applied University College: Morgan & Claypool Publishers. ISBN 9781681738932. Dostupné z: doi:10.2200/S01022ED1V01Y202006CSL011.
2. BUTTFIELD-ADDISON, Paris, Jon MANNING a Tim NUGENT. Unity Game Development Cookbook: Essentials for Every Game. O'Reilly Media, 2019. ISBN 978-93-5213-818-0.
3. JUDMAYER, Aljosha, Nicholas STIFTER, Katharina KROMBHOLZ a Edgar WEIPPL. Blocks and Chains Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms. Morgan & Claypool Publishers, 2017. ISBN 9781627057165. Dostupné z: doi:10.2200/S00773ED1V01Y201704SPT020.
4. SANTANA ROLDAN, Carlos. React 17 Design Patterns and Best Practices: Design, build, and deploy production-ready web applications using industry-standard practices. 3rd. Packt Publishing, 2021. ISBN 978-1800560444.
5. Ethereum [online]. Ethereum, © 2022 [cit. 2022-12-01]. Dostupné z: https://ethereum.org/en/.

Vedoucí diplomové práce: **Ing. Tomáš Vogeltanz, Ph.D.**
Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce: **2. prosince 2022**
Termín odevzdání diplomové práce: **26. května 2023**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 7. prosince 2022

Jméno, příjmení: Bc. David Rábel
Název diplomové práce: Vývoj Web3 NFT herní aplikace
Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Bc. David Rábel, v.r.
podpis studenta

ABSTRAKT

Tato diplomová práce se zaměřuje na vývoj herní aplikace s využitím Web3 a Non-Fungible Tokens (NFT). Cílem práce je analyzovat současné podmínky a vytvořit optimální prostředí pro vývoj a implementaci takové aplikace. Práce obsahuje teoretický přehled klíčových aspektů, včetně webového vývoje a popisu fungování Ethereum Virtual Machine (EVM). Praktická část se věnuje vývoji dvou webových aplikací, vytvoření a implementaci databáze a čtyřvrstvého backendu pro zpracování a ukládání hráčských transakcí. Dále práce popisuje vytvoření a implementaci NFT kolekce ve vhodném formátu pro blockchain hru. V závěru je v práci zahrnuto vytvoření Unity asset package, který poslouží jako základ pro budoucí vývoj kompletní aplikace.

Klíčová slova:

NFT, Web3, Unity, Token, Blockchain, EVM, React, Web, Webová aplikace

ABSTRACT

This thesis focuses on the development of a game application using Web3 and Non-Fungible Tokens (NFT). The aim of the thesis is to analyze the current conditions and to create an optimal environment for the development and implementation of such an application. The thesis contains a theoretical overview of key aspects, including web development and a description of the functioning of the Ethereum Virtual Machine (EVM). The practical part is devoted to the development of two web applications, the creation and implementation of a database and a four-layer backend for processing and storing player transactions. Furthermore, the thesis describes the creation and implementation of an NFT collection in a suitable format for a blockchain game. Finally, the thesis includes the creation of a Unity asset package that will serve as the basis for future development of a complete application.

Keywords:

NFT, Web3, Unity, Token, Blockchain, EVM, React, Web, Web application

Rád bych poděkoval svému vedoucímu práce panu Ing. Tomáši Vogeltanzovi, Ph.D., za jeho vedení, dobré rady, čas a velkou trpělivost, bez kterého by tato práce neměla šanci vzniknout.

Dále bych chtěl poděkoval také Bc. Janu Smržovi za pomoc a rady s blockchainovou částí.

Rád bych poděkoval své rodině a kamarádům za velkou podporu a trpělivost nejen při tvorbě této práce, ale i po celou dobu studia.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Vývoj herních aplikací na blockchainu je dnes obrovské technologické téma, které jde neustále kupředu a přináší nové možnosti jak pro vývojáře, tak pro hráče. S rostoucím zájmem uživatelů právě o tuto technologii, jsou herní aplikace jedním ze dvou odvětví blockchainu, které na trhu ukazují největší konzistenci, a to i v případě, kdy tomu trh nejde naproti.

Existuje mnoho firem a vývojářů, kteří se snaží zaměřovat na vývoj právě herních aplikací na blockchainu, a to právě z toho důvodu, že je zde obrovský potenciál, možnost implementace nových způsobu přístupu ke hrám a zároveň se zde nachází i obrovská tržní hodnota. Je však důležité podotknout, že tato technologie, přináší mnoho překážek a složitých problémů, kvůli kterým většina těchto herních aplikací na blockchainu po nějaké době od vydání upadne a ztratí zájem hráčů.

Tato práce má za cíl navrhnout a vytvořit vhodné prostředí pro NFT herní aplikaci. A to takovým způsobem, že nejprve bude navržena NFT kolekce ve vhodném EVM standardu. Poté by měla být vytvořena webová aplikace, která bude umožnovat uživatelům získávání nových tokenů. Následovat bude informativní webová stránka, která bude sloužit uživatelům k tomu, aby zjistili informace o herní aplikaci. Dále má tato práce za cíl vytvořit vhodné databázové prostředí pro uložení NFT dat a dat uživatelů, včetně ukládání dat všech transakcích na blockchainu týkající se hry. Součástí práce také je, backendové řešení, které bude přistupovat k vytvořené databázi, a to pomocí čtyřvrstvé architektury konkrétně zamřené právě na repository pattern a databázové modely. Nakonec bude vytvořen základní unity asset package, které budou sloužit pro následný vývoj aplikace.

Přínosem práce je připravení základního prostředí pro budoucí vývoj inovativní NFT herní aplikace, která bude přitahovat uživatele na základě unikátních mechanik a propojení s blockchainem. Mezi toto prostředí, lze zařadit vytvoření NFT herních kolekcí, které budou poskytovat pevný základ pro fungování herní aplikace a herních předmětů. Databázové a backendové řešení, které je robustní a efektivní pro správu všech potřebných dat. Webové aplikace, které budou sloužit jak pro rozvoj komunity, tak snadnou dostupnost informací novým i stávajícím uživatelům. A následně i jednoduchost řešení, kterou právě přináší základní unity asset package, který umožní snadný vývoj a rozšíření aplikace.

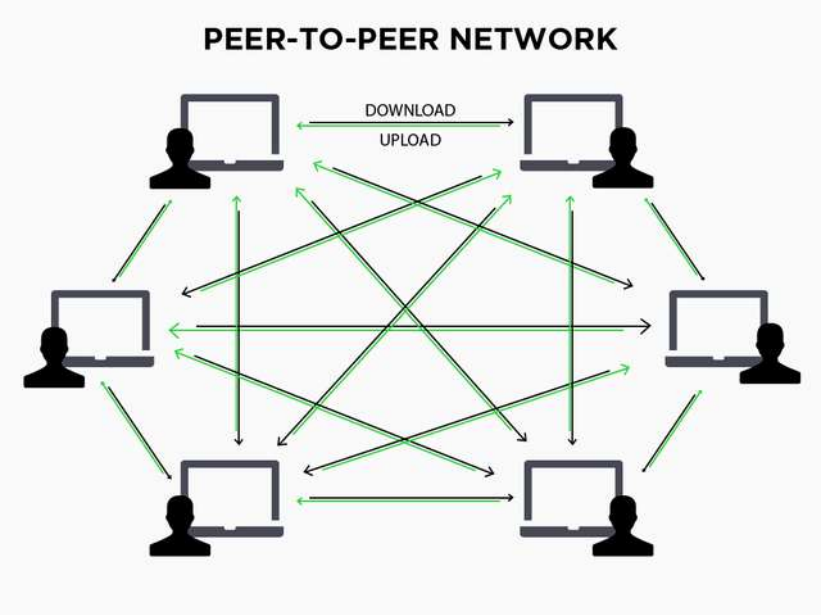
I. TEORETICKÁ ČÁST

1 BLOCKCHAIN

Blockchain se dá považovat za distribuovanou účetní knihu. U obyčejné účetní knihy, kterou známe z běžného světa je nutnost třetí strany pro ověřování plateb. Dále je potřeba notář, který je využit k ověřování transakcí a úschovu o různých hlasováních. S blockchainem se od těchto všech věcí odkláníme, a snažíme se co nejvíce omezit tyto aspekty centralizace. Centralizace může být sama o sobě dobrá věc, ale často není efektivní, a to hlavně díky vysokým nákladům, možnostem korupce se ztrátou soukromí. [1]

Pokud se bavíme o blockchainu, je považován za distribuovanou síť, kde právě tyhle věci odpadají, protože o výsledku a platnosti transakcí rozhodují jejich účastníci. [1]

Blockchain je založen na P2P (Peer to Peer), který v jednoduchosti funguje tak, že se jedná o síť počítačů a zařízení, které jsou nazývané jako peer a sdílí mezi sebou veškeré informace. Každý peer, je si roven a každý má své práva a povinnosti. Peer je jak klient, tak server zároveň. Dá se tedy považovat za síť, která nemá žádný centrální prvek jako je například server. Obrázek 1 popisuje právě tohle propojení. [2]



Obrázek 1 Peer to Peer network [56]

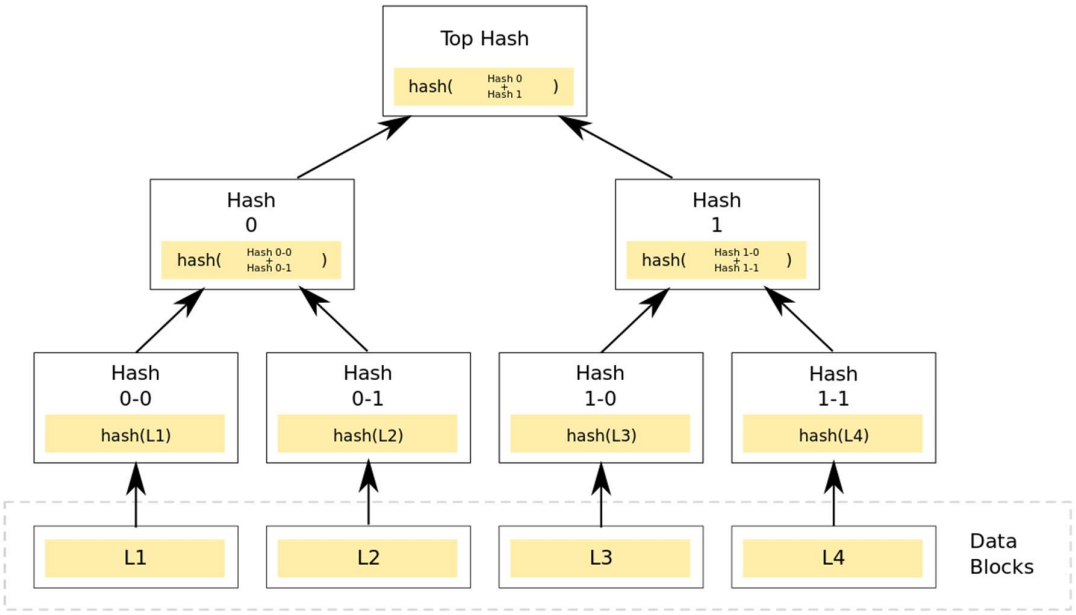
Za velkou výhodu těchto sítí, lze považovat to, že když je jeden peer nefunkční, ostatní stále fungují a komunikují mezi sebou, tím pádem shodit takovou síť, která má velký počet peeru

je prakticky nemožné. Zároveň je tato sít také velmi dobře škálovatelná a přidávání nových peeru je velmi jednoduché [2].

Další podstatnou částí blockchainu, kterou je nutno pochopit je asymetrická kryptografii. Je to soubor dvou klíčů, a to veřejného a privátního. U většiny implementací blockchainu, je právě veřejný klíč využit jako veřejná adresa uživatele, na kterou poté přijímá různé kryptoměny a jiná aktiva. Zatímco soukromý klíč, je využit jako „heslo“ k účtu. [1]

Poslední částí, která je pro blockchain jako takový důležitá, je metoda hashovaní. Jedná se proces, který vrátí jedinečný identifikátor pro danou sadu dat. Lze říct, že je to takový digitální otisk. Hlavní funkcionalitou hashovaní je to, že pokud změníte pouze jedno číslo, písmeno, symbol anebo jakoukoliv část dat, bude výsledný hash vypadat jinak. Tímto je zajištěno, že data nebyla změněna, ani jinak manipulována. Ve světě blockchainu se využívá proto, aby bylo možné rychle porovnávat a ověřovat velké množství dat. Blockchain sám o sobě tuto metodu využívá pro svoje základní fungování. Konkrétně využívá hashovaní zvané Merkle tree. [3]

Merkle tree, je datová struktura, která zajišťuje způsob toho, jak ukládat velká množství dat. Je to vlastně obrácený strom, kde každý list, je brán jako uzel, a každý uzel reprezentuje hash hodnotu dat. Pro ověření integrity dat stačí ověřit pouze hash hodnotu kořene stromu, což je výsledná hash hodnota celého stromu. Pokud se změnila alespoň jedna hash hodnota v některém z uzlu stromu, výsledná hash hodnota kořene se také změní, což signalizuje změnu dat a manipulaci. Dá se tedy shrnout, že je to v podstatě hashovací metoda, která umožnuje efektivně ověřit, zda data nebyla manipulována. Obrázek 2 ukazuje strukturu merkle tree [1][3]



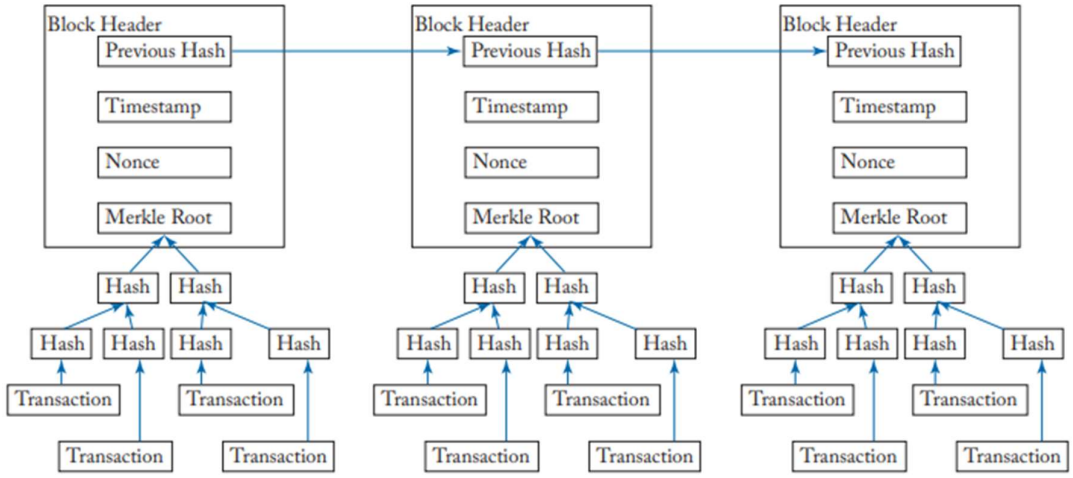
Obrázek 2 Merkle Tree [57]

Všechny tyto části, nám formují kompletní strukturu blockchainu, díky čemuž je tato technologická záležitost schopná fungovat. [1][3]

1.1 Vytváření nových bloků

Blockchain je vlastně jen strukturou bloků, které jsou seskládány za sebe a jsou vzájemně propojeny. Tyto bloky obsahují data ohledně všech akcí, které byly a jsou na blockchainu provedeny. Jsou vzájemně propojené a obsahují vždy i hash hodnotu předchozího bloku. Tato hash hodnota je vždy využita pro spočítání nového bloku. Zároveň však tato hash hodnota je využita i jako ověření, že s blokem nebylo manipulováno. Důležité pravidlo je to, že rodič může mít více potomků, zatímco potomek může mít pouze jednoho rodiče. V případě, že nastane situace, kdy je více potomků, nastává fork a do systému pokračuje pouze jeden blok. [1][3]

Důležité je, aby každý blok obsahoval předchozí hash hodnotu, a také časový údaj, kdy byl blok vytvořen. Dále by měl obsahovat *nonce* a *Merkle root*, který je zde využit k uložení všech transakcí a akcí, které se v daném bloku udály. Obrázek 3 ukazuje tento proces vytváření a jeho popis. [1][3]



Obrázek 3 Vytváření bloků [58]

Proces, jakým se dostávají nové bloky do systému, kdo má právo na to je vytvořit, a kdo rozhodne jaké transakce do bloku umístí, se nazývá mining. Tento proces se mění dle mechanismu konsenzu. [1]

1.2 Mining

Jedná se o jeden z nejdůležitějších konceptů blockchainu. Je to způsob, jakým se určuje, jaké transakce budou zařazené do nového bloku, a kdo tento blok bude vytvářet. Tato akce probíhá na předem domluveném konsenzu, díky čemuž se předchází jakýmkoliv podvodům. Těžba nového bloku je sama o sobě dosti nákladná a je třeba tohoto těžaře nějak odměnit, aby se mu pokryli náklady. Z tohoto důvodu, zde vznikají odměny pro těžaře. Tato odměna pro těžaře je bud určená u každé transakce, a tudíž jí určují uživatelé, co chtějí svoji transakci v dalším bloku, anebo je určená na základě algoritmu. Poté co je blok vytěžen, mohou si všichni účastnici blockchainu zkontrolovat jeho platnost na základě hash hodnoty a daného konsenzus mechanismu. [1][3]

1.3 Proof of work

Jedná se o nejvíce rozšířenou mechaniku konsenzu v blockchain světě. Je to vlastně způsob, jak využít velké množství výpočetní síly a energie k vytvoření bloku pomocí řešení složitých matematických rovnic. K vytvoření bloku se využívá speciálního algoritmu (SHA-256 v případě Bitcoinu). Těžaři, kteří tedy chtějí získat tento blok pro sebe se snaží najít hash

hodnotu pro všechna data v daném bloku tak, že přidávají celá čísla, dokud nenajdou taková, kde je prahová hash hodnota nižší než určitá číselná hodnota. Prahovou hodnotu lze měnit tak, aby se určila obtížnost problému. Tím se řídí pravděpodobnost uhodnutí správného výsledku, a proto je zajištěno, že doba mezi bloky zůstane zhruba stejná. Jakmile nějaký těžař nalezne řešení tohoto algoritmu, dostane těžařskou odměnu a poté začne stejný proces s novým blokem. Nejznámějším blockchainem, který tento konsenzus využívá je Bitcoin. [3][4]

Mezi výhody se řadí, hlavně bezpečnost, zatímco mezi nevýhody tohoto konsenzu lze zařadit drahá vstupní investice do mining hardwaru, ale hlavně obrovská energetická náročnost. [4]

1.4 Proof of stake

Tato metoda konsenzu, odstraňuje problém energetické náročnosti, ale zhoršuje částečně bezpečnost. Proto, aby se mohl uživatel zapojit do této metody potřebuje vlastnit nějakou část hodnoty na daném blockchainu a potřebuje tuto hodnotu zamknout do sítě. Šance na to, že vyhraje možnost vytvořit nový blok, je závislá dle velikosti hodnoty, kterou uložil. Tímto však vzniká problém, že nejbohatší uživatelé se stávají bohatšími. Tento problém je u každého proof of stake konsenzu řešen trochu jinak. Nejčastěji však nějakou procentuální pravděpodobností na vybrání náhodného uzlu. Jako příklad proof of stake mechanismu, lze považovat blockchain Ethereum. [3][4]

Mezi výhody se řadí hlavně nízká energetická náročnost a levnější vstupní investice k tomu, aby se uživatel zapojil se do sítě jako validátor. Mezi jeho hlavní nevýhodu lze brát v potaz tzv 51% útok, kde v případě, že by se jeden uzel získal 51 % všech prostředků, je schopný ovlivnit veškerá dění na blockchainu a převzít nad ním kontrolu. [4]

1.5 Privátní Blockchainy

Za privátní blockchain se dá považovat takový blockchain, do kterého se mohou připojit uživatelé jen na základě soukromé pozvánky, kteří jsou předem ověření a autentizováni. Ověření provádí provozovatelé sítí, a to prostřednictvím chytrých smluv, anebo jiných automatizovaných metod schvalování. [5]

Soukromé blockchainy vždy kontrolují, kdo se smí a nesmí do sítě zapojit. Pokud privátní síť umožnuje těžení a ověřování bloků, může vlastník (skupina vlastníků) soukromého blockchainu kontrolovat, kteří uživatelé mohou provádět těžbu a určovat jejich velikost

odměn. Kromě toho, mohou rozhodovat o tom, kdo bude sdílenou účetní knihu spravovat. Také mohou přepisovat, upravovat a mazat potřebné záznamy v blockchainu dle potřeby a svého uvážení. [5]

1.5.1 Výhody

Soukromý blockchain není decentralizovaný a tím pádem funguje jako zabezpečená databáze moderními kryptografickými koncepty, určenými pro potřeby organizace. Pouze ti lidé, kteří mají oprávnění mohou spustit nový uzel (node) a provádět, ověřovat a měnit blockchain. [5]

Právě díky, tomuto konceptu je možné node pouštět pouze pro specifickou organizaci. Tahle výhoda je důležitá pro větší průmysl, kde anonymita není úplně vítána. [5]

1.5.2 Nevýhody

Přestože jsou privátní blockchainy navržené pro podnikové aplikace, ztrácejí tak mnoho cenných vlastností, které poskytují veřejné blockchainy. Místo toho, aby blockchain byl plně rozšiřitelný, je vytvořen specifiky pro splnění konkrétních úkolu a požadavků průmyslu. [5]

Nedá se vždy považovat za plně bezpečný, protože zde existuje mechanismus konsensu k dosažení shody o transakcích a datech a ten má dost často omezený počet ověřovatelů, který nebývá vždy velký. [5]

Také ne všechny privátní blockchainy, mají konsensus o neměnnosti dat, tím pádem může provozovatel a správce data měnit. [5]

1.6 Veřejné Blockchainy

Za veřejný blockchain se dá považovat takový blockchain, do kterého se může kdokoliv volně připojit a podílet se na hlavních činnostech blockchainové sítě. Kdokoliv může číst a zapisovat. Zároveň může také každý kontrolovat probíhající činnosti ve veřejné blockchainové síti, díky čemuž je právě podtržena podstata samosprávné decentralizace. [5]

Za nejznámější veřejné blockchainy lze považovat Bitcoin, Ethereum, Cardano a Solanu. [5]

1.6.1 Výhody

Většina veřejných blockchainů, funguje na základě motivačního schématu, které podporuje nové uživatele, aby se zapojovali a udržovali sít. Veřejné blockchainy jsou zároveň plně decentralizované a poskytují bez autoritativní provoz.

Za výhodu lze také považovat to, že mohou sloužit jako páteřní sít pro jakékoliv decentralizované řešení. Za největší výhodu veřejných blockchainu se považuje bezpečnost, a to z důvodu, že zde platí poučka čím více účastníků, tím je blockchain bezpečnější. [5]

1.6.2 Nevýhody

Jako hlavní nevýhodu, lze určit velká spotřeba energie, která je potřeba k jejich údržbě. Je to způsobeno hlavně tím, že je většinou použit mechanismus konsensu, který vyžaduje, aby účastnici soutěžili o potvrzení informací a získali odměnu za to, že nechají sít využít svůj výpočetní výkon (Proof of work). Avšak ne všechny veřejné blockchainy mají tuhle nevýhodu, některé využívají jiný mechanismus (Proof of stake), které nejsou tak energeticky náročné, ale přinášejí jiné nevýhody. [5]

Mezi další nevýhodu pro část uživatelů, lze považovat nedostatek anonymity a soukromý, jelikož veřejné blockchainy umožnují komukoli nahlédnout do výše transakcí a adres, kterých se transakce týkají. Pokud se tudíž vlastnící adres stanou známými, ztrácí tak úplnou anonymitu. [5]

1.7 Konsorcium Blockchainy

Tyto blockchainy, jsou kombinací veřejných a privátních blockchainu a nabízí mnoho způsobu adaptací na situaci. [5]

1.7.1 Výhody

Jako jednu z hlavních výhod lze považovat, možnost připojení nových uživatelů, na základě automatizovaného procesu ověření totožnosti například KYC. Také je možné povolit se připojit každému, ke čtení ale zároveň povolit zápis jen v případě úspěšného ověření. Tyto blockchainy nabízejí širokou škálu funkcí. Nejpopulárnější jsou dnes pro podniky Blockchain-as-a-Service (BaaS), které byly navrženy tak, aby byly škálovatelné pro potřeby klienta, a mohli tak být dodávány na klíč. Právě díky tomu, že je blockchain dodáván jako služba, lze snížit náklady na vytvoření takového blockchainu pro společnosti. [5]

1.7.2 Nevýhody

Za nevýhody těchto blockchainu, lze vlastně považovat většinu problémů, které mají jak veřejné, tak privátní blockchainy. Největší nevýhodou je však ta, že tyto blockchainy jsou založeny na bezpečnostních opatřeních pro schvalování nových lidí, tudíž jsou to opatření, která lze teoreticky obejít například nabouráním se do učtu výše postavených uživatelů, čímž je pak možné se dostat k soukromým informacím. Podobný případ se stal v roce 2013 kdy byla napadena společnost Target a vzhledem k tomu, že měla přistup do sítě, byly skrze ni ukradena i jiná data firem. [5]

1.8 Historie

První zmínku o blockchainu, můžeme vidět už v roce 1982 od David Chaum, který udělal svoji disertační práci, která měla název „Computer Systems Established, Maintained, and Trusted by Mutally suspicous Groups“. Další zmínka byla v roce 1991 od Stuarta Habera a W. Scott Stornetta, kteří chtěli vytvořit systém, kde by se dali ukládat dokumenty s časovým razítkem bez toho, aniž by museli být fyzicky označeny. V roce 1992 Haber, Stornetta a Dave Bayer přišli s mechanismem Merkle Tree, kde razantně zlepšili efektivitu a zavedli myšlenku, jak uložit několik dokumentů obsahujících certifikát do jednoho bloku. V roce 1995 byly také jejich hashe certifikátů, pravidelně zveřejnovány ve slavném mediu New York Times. [6]

Blockchain, tak jak ho známe dnes byl vytvořen osobou anebo skupinou osob používající pseudonym Satoshi Nakamoto v roce 2008, aby sloužil jako veřejná distribuovaná účetní kniha pro transakce kryptoměny Bitcoin na základě předchozí práce Stuarta Haberta, W.Scotta Stornetta a Davea Bayera. Právě implementace Bitcoinu z něj udělala první digitální měnu, která řešila problém dvojího utrácení bez potřeby důvěryhodné autority anebo centrálního serveru. Právě konstrukce Bitcoinu inspirovala další aplikace a blockchain, které jsou v dnešní době populární [7]

V roce 2014, byla velikost všech bloků na Bitcoinu 20 GB, v roce 2015 30 GB, v roce 2016 až 2017 tahle velikost narostla z 50 GB na 100 GB a k roku 2023 již je tato velikost 435 GB. [8]

2 ETHEREUM VIRTUAL MACHINE (EVM)

Ethereum Virtual Machine, často označovaný zkratkou EVM, byl poprvé definován Gavinem Woodem, který ho popsal ve svém návrhu pro vylepšení sítě Ethereum. [1]

EVM je 256bitový zásobník registrů navržený tak, aby umožňoval spouštění kódu definovaného v chytrých smlouvách. Vytváří tedy vrstvu mezi kódem a strojem, což umožňuje oddělit aplikaci od hostitele. [1][9]

Tento virtuální stroj využívá tzv. chytrých smluv, které se nejčastěji programují v jazyce Solidity. EVM přijímá tyto chytré smlouvy. EVM má jednu velkou limitaci, která se dá považovat, jak za výhodu, tak i nevýhodu a to takovou, že je navázán na využití tzv. gas, který se platí za výpočetní výkon. Tento gas, však zároveň udržuje EVM od toho, aby se zacyklilo a nikdo nemohl vytvořit nekonečnou smyčku. Samotné chytré smlouvy EVM nezvládne zkompilovat, a proto musí být převedeny do nízko úrovňových instrukcí, které nazýváme operační kódy. Díky všem těmto vlastnostem je EVM Turing-complete. [1][9]

Každý operační kód zabírá jeden byte a může zde existovat pouze 256 operačních kódů, které jsou rozdělené do specifických kategorií, kde každý vykonává nedílnou součást systému. Jedná se o operační kódy, které manipulují s pamětí, operační kódy, které manipulují s úložištěm, operační kódy pro proměnné, operační kódy pro čítače programu, operační kódy pro zastavování operací a aritmetické, bitové a porovnávací operační kódy. [1][9]

Z důvodu efektivity jsou všechny tyto operační kódy uložené v byte kódech. Když tedy EVM provádí operaci, čte byte kód, kde každý byte reprezentuje nějaký operační kód. [1][9]

EVM tedy zvládne manipulovat pouze se 16 nejnovějšími položkami. Z tohoto důvodu operační kódy využívají paměť smart kontraktů. Je však dobré podotknout, že toto je pouze dočasná paměť a pokud chceme ukládat data na neomezenou dobu, je třeba využít storage (vnitřní paměti), která je však velmi drahá na právě zmíněný gas. [1]

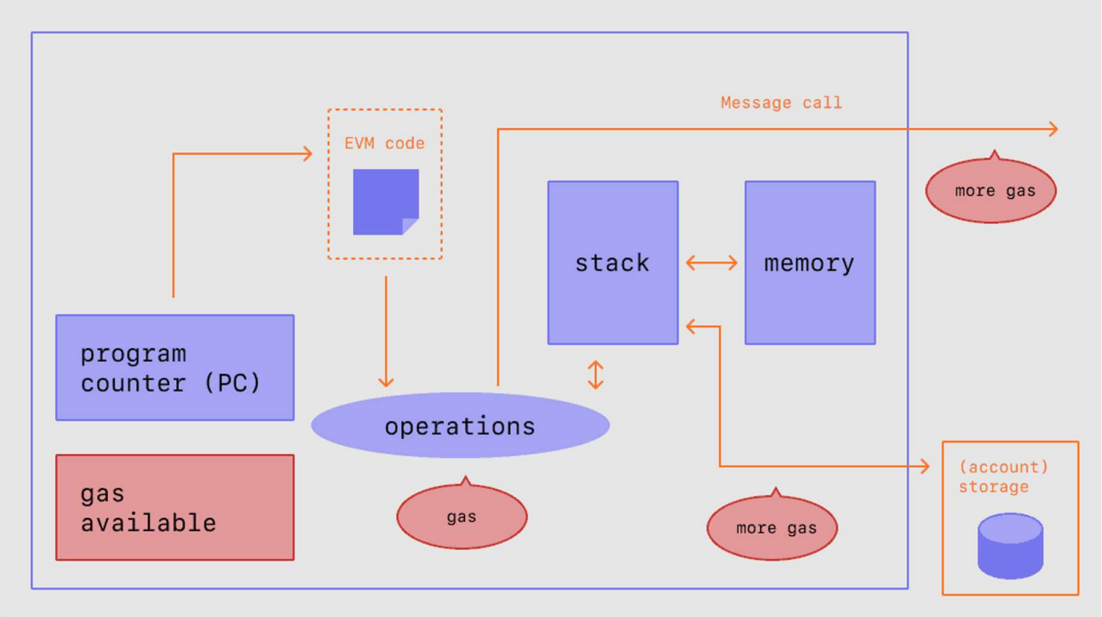
EVM se dále tedy chová jako klasický počítač, který má svoji smyčku událostí a kdykoliv na nějakou narazí, tak se ji snaží vykonat. Každá změna, kterou stroj provede, vychází z předchozího stavu, čímž zajistí to, že tyto změny na sebe budou navazovat. [1]

EVM na rozdíl od běžných počítačů kontroluje, zda nedochází k neoprávněným transakcím, a stav stroje je reprezentován zůstatkem, který v daném okamžiku existuje. Tento přístup

ovšem přináší také mnoho nevýhod, například to, že existuje mnohem více neplatných stavů než těch platných. [1]

Díky všem těmto událostem má tedy EVM kompletní historii, která ukazuje každou oprávněnou změnu stavu, která na něm proběhla. [1]

Funkcionalitu EVM popisuje Obrázek 4.



Obrázek 4 EVM [59]

2.1 Non fungible token (NFT)

NFT je zkratka pro non fungible token (nezaměnitelný token), jedná se o jedinečný digitální identifikátor, který nelze zkopírovat, nahradit nebo rozdělit. Je uložen na blockchainu a využívá se k potvrzení pravosti a vlastnictví digitálního aktiva a konkrétních práv k němu. [10]

V současné době, tento token reprezentuje hlavně digitální část umění a sběratelskou hodnotu, ovšem má mnoho jiným využití. NFT nám umožní tokenizovat věci, jako jsou umělecká díla, sběratelské předměty, anebo dokonce i nemovitosti. [11]

Za velkou výhodu NFT můžeme považovat to, že nikdy nevytvoříme dvě stejná NFT, vždy se budou v něčem lišit například v čase vytvoření. Zároveň každé NFT musí mít vlastníka a

ten je veřejně dostupný, díky čemuž si tuhle informaci mohou ostatní uživatelé snadno ověřit. Také jsou kompatibilní s jakýmkoliv EVM, tudíž v případě, že bychom vytvořili například vstupenky na koncert jako NFT, můžeme je prodávat na již existujících marketech a nemusíme vytvářet celou infrastrukturu pro tyto vstupenky. Také si mohou autoři těchto NFT, definovat autorské poplatky a pokaždé co někdo jejich NFT prodá, tak z toho získají podíl jako originální autoři. [11]

Už jenom z výhod nám vyplívá to, že NFT v dnešním světě mohou mít mnoho využití, mezi které se například řadí, unikátní digitální kolekce, unikátní herní předměty, doménová jména, vstupenky na koncert anebo třeba digitální certifikáty a digitální identity. [11]

2.1.1 První NFT

Historie NFT, lze dohledat již v roku 2012, když vyšel článek Meniho Rosenfielda, kde představil koncept „barevných mincí“ pro blockchain Bitcoin. Myšlenka spočívala v popisu tříd a metod, které sloužili pro reprezentaci a správu reálných aktiv na blockchainu za účelem prokázání vlastnictví těchto aktiv. Ovšem to, že tento koncept byl vytvořen na Bitcoinu způsobilo to, že z technických důvodu nemohl být nikdy realizován, avšak zasadil základ, ze kterého vzniklo NFT, které známe dnes. [12]

2.1.2 Nejznámější NFT na Ethereum blockchainu.

Za nejznámější NFT kolekci na Ethereum blockchainu se dá považovat kolekce Bored Ape Yacht Club (BAYC), která má k 29.11 2022 hodnotu 861 000 ETH. Tato kolekce čítá 10 000 unikátních obrázků znuděných opic, které byly využité jako vstupenky do VIP klubu. Nejdražší kus byl prodán za 3,408,000 $ viz Obrázek 5. [13]

Obrázek 5 Bored Ape #8817 [60]

2.2 Fungible tokeny (FT)

Fungible token neboli FT je token je zaměnitelný token a jeho hodnota je stejná jako hodnota jiného fungible tokenu. Jako příklad si můžeme vzít ETH token, kde 1 ETH lze zaměnit za 1 ETH a jeho hodnota bude vždy stejná, obvykle je FT využíván pro fiat měnu, anebo třeba pro hlasovací právo. Zjednodušeně řečeno u fungible tokenu záleží na tom, kolik ho máte a u NFT záleží jaký token máte. [14][15]

2.3 Nejznámější EVM Blockchainy

Za EVM blockchainy, můžeme považovat takové blockchainy, které jsou kompatibilní s EVM. Tj jedná se o blockchain, který je schopen provozovat EVM a zároveň překládat chytré smlouvy. V praxi to znamená, že vývojáři mohou psát a nasazovat stejnou chytrou smlouvu na všechny EVM kompatibilní blockchainy. Tyto smlouvy poté mohou komunikovat mezi sebou skrze různé blockchainy pomocí mostů.[16]

Mezi nejznámější EVM blockchainy můžeme zařadit Ethereum (ETH), který je otcem všech ostatních EVM blockchainu a využívá pro operace platidlo Ethereum. Právě na něm vznikla celá myšlenka a všechny ostatní blockchainy, které jsou kompatibilní, vychází z jeho implementace. [16][17]

Dalším velice známým EVM blockchainem, je Binance Smart chain (BNB), který vlastní největší kryptoměnová směnárna Binance. Tento blockchain využívá Binance coin, pro operace, které na něm probíhají. [16][18]

Mezi další velké EVM blockchainy patří také Polygon, který je asi nejvíc podobný právě zmiňovanému Ethereum blockchainu na rozdíl od ně, však pro své účely využívá měnu MATIC. Je také velmi známý velkým počtem vývojářů, velkou komunitou a nízkými poplatky za využívání (GAS fee). [16][19]

Aktuální počet blockchainů, které podporují EVM je velké množství, avšak ne všechny jsou, dostatečně známé anebo přináší unikátní případy užití, které by nalákali nové vývojáře anebo nové uživatele. [16]

3 ETHEREUM STANDARDY

Za Ethereum standard se dá považovat sada pravidel, které musí chytrá smlouva splňovat, tak, aby její základ měl vždy stejnou kostru, čímž dochází k tomu, že je kompatibilní se službami třetích stran. Tyto standardy jsou vlastně specifikace na úrovni aplikace, tak stejně jako jsou jednotlivé knihovny, popřípadě registry. Díky těmto jasně definovaným parametrům, může kdokoliv s dostatečnými znalostmi vytvořit vlastní ERC token. [20]

Nový standard může vytvořit kdokoliv, avšak je vyžadováno projít procesem návrhu na vylepšení Etherea (Ethereum Improvement Proposal neboli EIP), což je dokument s navrhovanými funkcemi a procesy pro blockchainovou síť Ethereum. Jakmile vývojář vytvoří svůj návrh, musí jej posoudit a zkontrolovat vývojáři Etherea. Poté přijde na řadu komunita, a pokud jej uzná za důležitou součást blockchainového ekosystému, bude návrh přijat, dokončen a implementován.[20]

3.1 NFT standardy

Za nejznámější NFT standardy jsou považovány ERC-721 a ERC-1155. Existují zde i další standardy, ale tyto standardy, popřípadě jejich deriváty jsou nejpoužívanější. [21]

3.1.1 ERC-721

ERC-721 standard je zaměřen pouze na non fungible tokeny (NFT). Nejčastěji se využívá pro reprezentaci digitálních sběratelských předmětů, digitálního umění, vstupenek na události atd. Pro využití tohoto standardu je nutno implementovat rozhraní Context, ERC165, a IERC721. Dále můžeme tento standard vylepšit pomocí implementace dalších rozhraní, jako je například ERC721 Metadata, popřípadě EC721 Enumberable. [22][23]

Každé NFT v tomto standardu, musí obsahovat tokenID (uint 256). Odkaz na uložiště, kde najdeme specifická data k NFT, jako jsou například metadata, které obsahují informace o tokenu, odkaz na obrázek atd. Dále může obsahovat další parametry, jako je maximální počet kusů, cena za jeden kus, popřípadě různá pole adres, která mohou sloužit pro další účely. [20]

Základní funkce vyžadovaného rozhraní

- **balanceOf(owner)**

  Tato funkce vrátí počet NFT na konkrétní adrese. [22][23]

- **ownerOF(tokenId)**

  Vrátí konkrétní adresu vlastníka, pro dané NFT. [22][23]

- **safeTransferFrom(from, to, tokenId)**

  Převede konkrétní NFT (dle token ID) z jedné adresy (from) na jinou adresu (to). Vyžaduje, aby parametr from, nebo to nebyla nulová adresa a aby tokenId byl tokenem, který adresa, co volá tuto funkci vlastnila, popřípadě byla privilegovaná pro posun konkrétního tokenu. [22][23]

- **transferFrom(from, to, tokenId)**

  Převede konkrétní NFT (dle token ID) z jedné adresy (from) na jinou adresu (to). Narozdíl od funkce safeTransferFrom nevyžaduje, aby parametr from anebo to nebyl nulový. [22][23]

- **approve(to, tokenId)**

  Schvaluje jinou adresu pro přenos daného tokenu. Zároveň však platí, že na jeden token může být v jeden okamžik pouze jedna schválená adresa. Tato funkce může být vyvolána pouze vlastníkem tokenu anebo schváleným operátorem. [22][23]

- **getApproved(tokenId)**

  Získá schválenou adresu, pro přesun tokenu, pokud adresa není nastavená vrátí 0. Pokud nastavená je vrátí ID tokenu. [22][23]

- **setApprovalForAll(operator, _approved)**

  Nastaví anebo zruší schválení operátora. Operátor následně může převést všechny tokeny z dané chytré smlouvy, které patří vlastníkovy. [22][23]

- **isApprovedForAll(owner, operator)**

  Vrátí true anebo false na základě toho, jestli je daná adresa schválená vlastníkem. [22][23]

- **safeTransferFrom(from, to, tokenId, data)**

  Bezpečně přesune vlastnictví daného tokenu dle ID na jinou adresu. Pokud je však cílovou adresou chytrá smlouva, je nutno mít implementovanou specifickou funkci. [22][23]

Základní eventy (požadované rozhraní)

- **Transfer(from, to, tokenId)**

  Event, který je spuštěn v případě, že se token přesune. [22][23]

- **Approval(owner, approved, tokenId)**

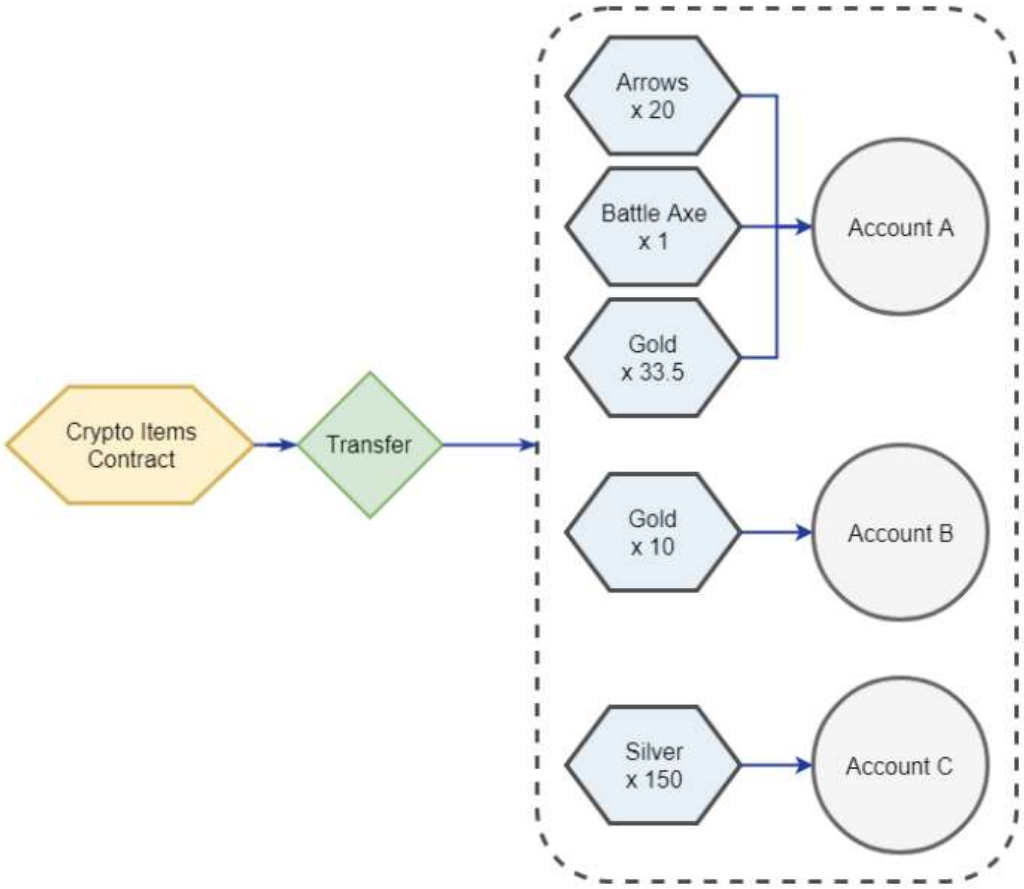
  Event, který je spuštěn v případě, že je schválená nová adresa pro přesunutí tokenů. [22][23]

- **ApprovalForAll(owner, operator, approved)**

  Event, který je spuštěn v případě, že je určen nový operátor. [22][23]

3.1.2 ERC-1155

Tento standard je vytvořen tak, aby využíval pouze jednu chytrou smlouvu pro reprezentaci více tokenů najednou. Proto se zde odlišuje i funkce BalanceOf, která má přidaný dodatečný argument. Dalo by se říct, že tento standard je velmi podobný ERC721, ale v tomto případě id tokenu nemá žádný vliv. Každé NFT zde bud existuje anebo nikoliv. Narozdíl od ERC721, kde balanceOf vrací kolik různých tokenů učet vlastní, tak ERC-1155 ukazuje rozdílný počet konkrétního tokenu. Tento standard, řeší problém toho, když projekt potřebuje více stejných tokenů. Namísto vytváření nového kontraktu, může jeden ERC-1155 udržovat stav celého systému, což snižuje náklady na vytváření chytrých smluv. Obrázek 6 popisuje fungování tohoto standardu. [24][25]



Obrázek 6 ERC1155 [61]

Základní funkce (požadovaný interface)

- **balanceOf(account, id)**

  Vrací množství tokenů typu tokenid vlastněných účtem [24][25]

- **balanceOfBatch(accounts, ids)**

  Vrátí skupinu tokenů [24][25]

- **setApprovalForAll(operator, approved)**

  Udělí anebo zruší povolení jiné adrese pro přenos tokenů [24][25]

- **isApprovedForAll(account, operator)**

  Vrátí hodnotu true, pokud je operátor schválen k převodu tokenů [24][25]

- **safeTransferFrom(from, to, id, amount, data)**

  Převede tokeny z jedné adresy na druhou [24][25]

- **safeBatchTransferFrom(from, to, ids, amounts, data)**

  Převede více tokenů najednou z jedné adresy na druhou [24][25]

Základní eventy (požadovaný interface)

- **TransferSingle(operator, from, to, id, value)**

  Event, který je spuštěn v případě, že se přesune pouze jeden token [24][25]

- **TransferBatch(operator, from, to, ids, values)**

  Event, který je spuštěn v případě, že se přesune více tokenů [24][25]

- **ApprovalForAll(account, operator, approved)**

  Event, který je spuštěn v případě, že je určen nový operátor [24][25]

- **URI(value, id)**

  Event, který je spuštěn v případě, když se změní URI pro typ tokenu. [24][25]

3.2 FT standardy

Za nejznámější FT standard je považován ERC-20. Tento standard zastává, prakticky většinu všech FT tokenů. Tento standard definuje základní pravidla chování tokenu a implementuje jeho převody a správu. [26]

3.2.1 ERC-20

Tento standard, je vytvořen pro fungible tokeny (FT). Tudíž jakýkoliv token se rovná tokenu jinému a vždy mají stejnou hodnotu. Žádné tokeny nejsou unikátní a ani nemají žádná specifická práva anebo unikátní chování. Díky tomuto jsou tokeny ve standardu ERC-20 užitečné pro vytvoření vlastní měny, finančního prostředku anebo třeba pro hlasovací právo. Viz Kapitola FT.[27][28]

Základní funkce (požadované rozhraní)

- **totalSupply()**

  Vrátí počet existujících tokenů [27][28]

- **balanceOf(account)**

  Vrátí množství tokenů vlastněných specifickou adresou [27][28]

- **transfer(recipient, amount)**

  Přesune specifické množství tokenů na jinou adresu [27][28]

- **allowance(owner, spender)**

  Vrátí zbývající počet tokenů, které může specifická adresa utratit za vlastníka [27][28]

- **approve(spender, amount)**

  Nastaví množství tokenů, které může za daného vlastníka daná adresa utratit. [27][28]

- **transferFrom(sender, recipient, amount)**

  Přesune tokeny na zadanou adresu pomocí allowance mechanismu. [27][28]

Základní eventy (požadované rozhraní)

- **Transfer(from, to, value)**

  Event, který se zavolá v případě, že se přesunou tokeny [27][28]

- **Approval(owner, spender, value)**

  Event, který se zavolá v případě, že majitel tokenů nastaví jinou adresu pro možnost utrácení jeho tokenů [27][28]

4 WEBOVÝ VÝVOJ

Webový vývoj je proces vytváření, budování a údržby webových stránek anebo webových aplikací, které běží online v prohlížecí, nebo na soukromé síti. Proces webového vývoje zahrnuje nejen webový design, ale také samostatný vývoj webu, jeho naprogramovaní, bezpečnost a stabilitu. Jedná se tedy o komplexní téma, které zahrnuje spousty různých oborů. [29][30]

4.1 Webová aplikace

Webová aplikace je počítačový program, ke kterému lze přistupovat pomocí webového prohlížeče Webová aplikace je také často rozeznávaná tak, že je vytvořená, aby na ní konečný uživatel vykonával interakce. Například Google Apps anebo Youtube. [31]

4.2 Webová stránka

Webová stránka je soubor relativních stránek, které obsahují texty, obrázky a zvuk. Poskytuje vizuální a textový obsah, který si uživatelé mohou prohlížeč a číst. Tak jako u webových aplikací je nutno webového prohlížeče, avšak u webové stránky se jedná pouze o informativní účel, bez nutné uživatelské interakce. Například restaurační weby pro prohlednutí jídelního lístku, fóra či blogy. [31]

Porovnání webové aplikace a webové stránky lze pozorovat na Tabulka 1.

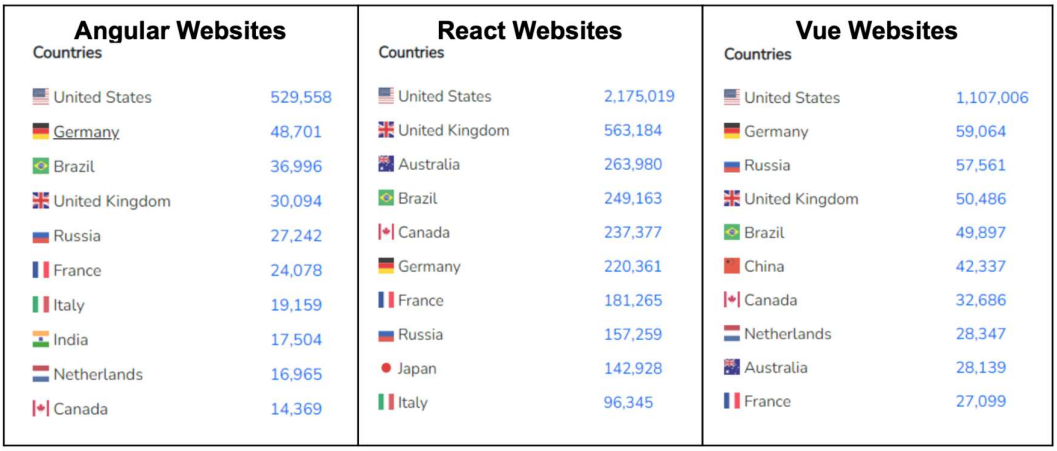
Tabulka 1 Webová aplikace vs webová stránka [64]

Webová aplikace	**Webová stránka**
Webová aplikace je určena pro interakci s koncovými uživateli.	Webové stránky obsahují v podstatě statický obsah.
Uživatel webové aplikace může číst obsah webové aplikace a také manipulovat s daty.	Uživatel webových stránek může pouze číst obsah webových stránek, ale nemůže s nimi manipulovat.
Webová aplikace by měla být před nasazením předkompilována.	Webové stránky nemusí být předem zkompilovány .
Funkce webové aplikace je poměrně složitá.	Funkce webových stránek je jednoduchá.
Webová aplikace je pro uživatele interaktivní.	Webové stránky nejsou pro uživatele interaktivní.
Webová aplikace většinou vyžaduje ověření	Na webových stránkách není ověřování nutné.

4.3 Moderní frameworky pro vývoj v JavaScriptu a TypeSriptu.

Za moderní webový framework můžeme považovat takový framework, který usnadňuje práci webovému vývojáři. Tento framework by měl mít za cíl urychlit a zefektivnit webový vývoj. [32]

Na trhu se nachází mnoho různých frameworků a každý přináší své výhody a nevýhody, avšak mezi ty nejznámější můžeme zařadit React, Angular, Vue.js, Boostrap, Materialize, Laravel. Obrázek 7 ukazuje počet spuštěních stránek v jednotlivých frameworcích [32][33]



Angular Websites

Countries	
United States	529,558
Germany	48,701
Brazil	36,996
United Kingdom	30,094
Russia	27,242
France	24,078
Italy	19,159
India	17,504
Netherlands	16,965
Canada	14,369

React Websites

Countries	
United States	2,175,019
United Kingdom	563,184
Australia	263,980
Brazil	249,163
Canada	237,377
Germany	220,361
France	181,265
Russia	157,259
Japan	142,928
Italy	96,345

Vue Websites

Countries	
United States	1,107,006
Germany	59,064
Russia	57,561
United Kingdom	50,486
Brazil	49,897
China	42,337
Canada	32,686
Netherlands	28,347
Australia	28,139
France	27,099

Obrázek 7 Počet stránek [62]

4.3.1 React

Jedná se o Javascriptovou knihovnu, která je vytvořena pro vývoj webů. Nejčastěji se využívá pro stavbu singlepage aplikací, a dovoluje nám vytváření znovu použitelných komponent. Využívá také virtuálního document object modelu (DOM), který právě vytváří virtuální kopii originálního DOM. Je také specifický tím, že využívá JSX soubory, které obsahují HTML jazyk a Javascript v jednom souboru, který je následně vložen do webové stránky. [32][34]

Je také dobré podotknout, že se jedná o deklarativní framework, to znamená, že není nutné přesně definovat, jak se má daná věc vykonat, ale stačí definovat co chceme. Díky tomuto přístupu se dá vývoj webové aplikace znatelně urychlit. [32][34]

Za výhodu tohoto frameworku, lze tedy považovat, jeho jednoduchost, možnost znovuvyužití komponent, jeho virtuální DOM. Naopak za nevýhodu lze považovat JSX, kde v případě nedodržení určitých standardů, může přinášet nepořádek v kódu a potencionální bezpečnostní hrozby do aplikace. [32][34]

4.3.2 Agular

Jedná o Typescriptový framework, který je udržován společností Google. Primárně tak, jako React je také navržen pro tvorbu single page aplikací. Využívá MVC architektury a rozděluje kód do modulů [32][35]

Velkou výhodou tohoto jazyka je, že je psaný v Typescriptu, čímž přináší silnou typovou kontrolu a zajišťuje tak větší bezpečnost. Díky strukturování aplikace, je tento kód také dobře testovatelný a škálovatelný, což může být pro spousty projektů klíčové. [32][35]

Naopak za nevýhodu lze považovat jeho složitost, právě kvůli zmiňovaným opatřením, je tento framework robustní a složitý. Na tento fakt navazuje také to, že velikost projektu v Angularu je vyšší a v určitých ohledech je pomalejší. [32][35]

4.3.3 Vue.js

Jedná se o Javascriptový framework, který je vytvořen pro tvorbu webových rozhraní. Je postaven na HTML, CSS a JavaScriptu a poskytuje deklarativní přístup. Také využívá přístupu stavby komponentů pro urychlení vývoje aplikace a pro vytvoření přehlednosti. Tyto komponenty skládá do stromové struktury, kde jsou komponenty jako větve. [32][36]

Stejně jako React také využívá Virtuálního dokument objekt modelu. [32][36]

Za výhodu frameworku se určitě dá považovat to, že je tento framework velmi jednoduchý a tím pádem nepřináší spousty nepotřebného kódu navíc, také je velmi rychlý a má velmi dobrou dokumentaci. Naopak za nevýhodu by se dalo považovat to, že není příliš vhodný pro větší projekty a snaží se být až příliš flexibilní. [32][36]

4.4 Web3 knihovny

Knihovny Web3 jsou sbírky nástrojů, funkcí a protokolů, které vývojářům usnadňují interakci s decentralizovanými technologiemi, jako jsou blockchainy a decentralizované aplikace a chytré smlouvy. Tyto knihovny vytváří abstrakci nad složitými technickými protokoly a usnadňují je vývojářům integrovat do svých projektů. [37]

Mezi nejznámější knihovny patří například knihovna web3.js, ether.js anebo například OpenZeppelin. [37][38]

4.4.1 Web3.js

Jedná se o nejpoužívanější Web3 knihovnu, která slouží pro připojení k veřejným a privátním EVM blockchainům. Poskytuje nezbytnou funkcionalitu pro prohlížeč a umožnuje nám vytvářet transakce a interakce s blockchainem. [38][39]

4.4.2 Ethers.js

Jedná se o javascriptovou knihovnu, která umožnuje interakci s Ethereum blockchainem a jeho ekosystémem. Původně tato knihovna byla navržena jen pro interakci s ethers.io ale nyní už má obecnější využití [38][39].

Velkou výhodou této knihovny je její velikost, která je 88Kb, také stojí za zmínku, že knihovna, má funkční testy a MIT licenci. [40]

4.4.3 OpenZeppelin

OpenZeppelin je opensource knihovna, která slouží, k vývoji chytrých smluv. Primárně implementuje standardy tokenu jako je ERC-20 a ERC-721.[40] [41]

4.5 Bezpečnost aplikace

Bezpečnost webových aplikací je ochrana webových stránek, aplikací a webových API, před kybernetickými útoky. Jedná se o složitou disciplínu, která má za úkol zajistit bezproblémové fungování webových aplikací a chránit je před krádeži dat a neetickými umysli s negativním důsledkem. [42]

Jeden z důležitých bodů zabezpečení aplikace je šifrování dat pomocí SSL certifikátů. Dalším z důležitých bodů je provést hodnocení rizik, které nastaví, bezpečnostní pravidla a minimalizuje bezpečností hrozby. [43]

Využití webového aplikačního firewallu (WAF), dokážu zajistit nebezpeční předtím, než se stane, a to právě tím, že přidává vrstvu mezi klient a server. [43]

5 BLOCKCHAIN HRY

Jedná se o hry, které využívají blockchain technologii pro svůj herní mechanismus. Existují zde hry, které využívají blockchain, pouze pro ověřovaní uživatelů, ale existují také hry, které využívají blockchain pro všechny své herní mechaniky. Blockchain může být u takových her využit pro ukládání herních předmětů, pro vytváření globálního tržiště, pro ověřovaní vlastnictví anebo pro vytváření vlastních fungujících společností uprostřed různých her. [44][45]

5.1 Rozdělení blockchain her

Hry bychom mohli rozdělit do následujících kategorií, dle jejich využití blockchainu. Je nutno brát v potaz, že toto rozdělení se liší, dle zdroje a subjektivity uživatele.[44]

5.1.1 Minimální využití blockchainu

Jedná se o hry, které běží na centralizovaných serverech. Tyto hry nejčastěji využívají pouze možnost kupování kosmetických předmětů (nejčastěji zastoupené jako NFT), které jsou navázány na blockchain tokeny. Díky této integraci, mohou hráči mezi sebou jednoduše tyto kosmetické předměty vyměňovat bez nutnosti integrovaní tohoto případu užití v samostatné hře. [44]

5.1.2 Střední využití blockchainu

V tomto případě se jedná o hry, které nabízí uživatelům ve hře generovat aktiva. Jedná se například o hry, které umožnují uživatelům skrze jejich NFT získávat vlastní tokeny, popřípadě jiné herní předměty, které poté uživatel může prodat na globálním tržišti. [44]

Přesto, že tyto hry také běží na centralizovaných serverech využívají tuto technologii již mnohem více než hry s minimálním využití. Tyto hru zavádějí takzvaný princip play to earn. [44]

5.1.3 Maximální využití blockchainu

Tyto hry využívají blockchain ve všech případech užití, které již byly zmíněny, avšak se liší tím, že jejich celá hra běží na blockchainu. Každá akce, kterou uživatel vykoná ve hře, musí být ověřena validátory. V konečném hledisku, to nejen znamená, že ve hře je prakticky nemožné podvádět, ale také to znamená, že vývojář ztrácí určitou kontrolu nad jeho hrou, což může být považováno za dvojsečnou zbraň. [44]

5.2 Tržní kapitalizace blockchain her

Tržní kapitalizace blockchainu je velké a důležité téma. Na začátku roku 2022, byla tržní kapitalizace blockchain her okolo 25 miliard dolaru a každým dnem, tato kapitalizace narůstá. Vzhledem k analýze DappRadaru, jsou na blockchainu dvě nejvíce stabilní odvětví, a to právě blockchain hry a NFT. [46][47]

Příkladem, který toto tvrzeni potvrzuje je velmi populární hra Axie Infinity, kde jeden jejich token měl i hodnotu 150 dolarů, která přinesla tržní hodnotu projektu, až na číslo 9 miliard dolarů a hráčskou základnu dvou milionů hráčů. [47]

Toto tvrzení také dokazuje fakt, že i přesto, že celý krypto svět šel do svého bear marketu, tak v prosinci 2021 hře Axie Infinity narostl počet hráčů až na tři miliony za den a aktivita na blockchainu se zvětšila skoro 4x. [47]

5.3 Nejznámější blockchain hry

5.3.1 Axie infinity

Za neznámější a nejúspěšnější hru na blockchainu se dá považovat hra Axie infinity, která má k dnešnímu dni volume za posledních 30 dnů 91,11 milionu dolarů. Jedná se o blockchain hru, která umožnuje hráčům sbírat, bojovat a rozmnožovat roztomilá zvířátka, kterým se říká Axies. Tyto zvířátka jsou reprezentované jako NFT. Tato hra kombinuje prvky virtuálního světa s prvky Tamagoči a karetní hry. Hru vydalo studio Sky Mavis, které aktuálně hru udržuje na dvou EVM blockchainem, a to na Ethereum a soukromém blockchainu Ronin. Ve hře se nachází dva herní ERC-20 tokeny, a to Smooth Love potion (SPL) a Axie infinity shards (AXS). [48][49][50]

Obrázek 8 ukazuje záběr herní mechaniky.

Obrázek 8 Axie bojový systém [63]

5.3.2 Decentraland

Za velice známou blockchain hru se považuje také Decentraland, jedná se o hru, která by se dala zahrnout do kategorie miniher, virtuální reality a virtuálního světa. Tato hra je založena na principu, že si lidé kupují virtuální pozemky pomocí jejich tokenu MANA, a na těchto pozemcích následně mohou stavět různé stavby, pořádat minihry anebo dělat třeba výstavy. Hra je primárně zaměřena na komunitní rozvoj a kreativitu uživatelů. [51]

Tato hra byla vydána v Březnu roku 2020 a to neziskovou organizací Decentraland Foundation. Tato hra je aktuálně spuštěna na Ethereum blockchainu.[52]

Obrázek 9 ukazuje jeden z eventů, který se ve hře může odehrávat.

Obrázek 9 Decentraland koncert uživatelů [64]

5.3.3 SandBox

Jedná se o velmi populární blockchainovou hru, která umožnuje hráčům vytvářet a sdílet minihry. Hra vychází se světa minecraftu a robloxu, které můžete znát z klasického web2 světa, tato hra navíc přidává obchodování s hernímy NFT. Tato hra by se dala popsat jako sandboxová simulace. [x] Byla vydána v roce 2021 a to společností Animoca Brands, primárním platidlem ve hře jsou SandBox tokeny tzv SAND. [53][54]

Hra je také velmi známá z důvodu uzavření spolupráci s velkými společnostmi, mezi které můžeme zařadit Atari, Skybound Entertaiment anebo třeba CoinMarketCap. [55]

II. PRAKTICKÁ ČÁST

6 TVOBRBA NFT KOLEKCE VE VHODNÉM STANDARDU

Tato kapitola se zaměřuje na tvorbu NFT (non-fungible token) kolekce s využitím vhodného standardu. Vzhledem k tomu, že NFT jsou digitální aktivum, je důležité zvolit takový standard, který umožňuje uživateli získat a spravovat NFT a současně ukládat metadata přímo na blockchain.

V první řadě je důležité provést analýzu požadavků, abychom zvolili nejvhodnější standard pro naši NFT kolekci a vybrali ten, který pro nás bude nejvýhodnější. Po výběru standardu probíhá implementace a deploy na testovací blockchain MUMBAI.

Následně se v této kapitole zaměřujeme na jednotlivé funkcionality smart contractu. To zahrnuje například vytvoření NFT, prodej NFT, způsob platby, převod NFT mezi uživateli, kontrolu vlastnictví NFT a další.

6.1 Analýza požadavků

Tato kapitola se zaměřuje na popis funkcionalit, které by měl smart kontrakt pro NFT kolekci poskytovat. Tyto funkcionality byly pečlivě vybrány s ohledem na to, co uživatelé očekávají od smart kontraktu pro NFT kolekce.

Funkcionality jsou navržené tak, aby uživatelé mohli snadno získávat a spravovat NFT v kolekci. Mezi tyto funkcionality patří například vytvoření NFT, prodej NFT, převod NFT mezi uživateli, kontrola vlastnictví NFT a další.

Důležitou součástí funkcionality smart kontraktu je možnost ukládání dat přímo na blockchain. To zaručuje, že metadata NFT jsou vždy přesné a aktualizované. Nejen, že je to pro uživatele lákavější, protože to podporuje decentralizaci, ale hlavně není vytvořena závislost na třetí straně.

Whitelist pro uživatele

Tato funkce by měla umožnit zápis uživatelů na whitelist pomocí merkle stromu. Uživatelé by měli být registrováni pomocí své adresy, která je následně podepsána adresou vlastníka. Následně by byl vytvořen listproof pro každého uživatele, který by sloužil k ověření, zda je uživatel na whitelistu. Pro ověření by byl použit vlastník kolekce, respektive adresa vlastníka, a právě tento podpis pro ověření. Tato funkce by měla být navržena tak, aby byla bezpečná a zajišťovala, že pouze oprávnění uživatelé mají přístup k NFT tokenům na whitelistu.

Transfer NFT

Funkce Transfer NFT umožňuje uživatelům přenést NFT tokeny z jedné adresy na druhou. Tato funkce je důležitá pro obchodování a převody vlastnictví NFT tokenů mezi uživateli. Transfer NFT by měl být zabezpečen tak, aby nedocházelo k neoprávněným převodům a krádežím tokenů. Uživatel by měl být schopen zadat adresu, na kterou chce NFT token převést, a transakce by měla být ověřena a potvrzena na blockchainu. Funkce Transfer NFT by měla být navržena tak, aby byla snadno ovladatelná a uživatelsky přívětivá. Uživatel by měl mít možnost rychle a snadno převést své NFT tokeny na jinou adresu.

Možnost si vybrat jaké NFT si uživatel bude mintovat

Tato funkcionalita umožňuje uživatelům vybrat si, které NFT tokeny budou vytvořeny a přidány do sbírky. Uživatel by měl mít možnost vybrat si požadovaný druh NFT tokenu pomocí číselného kódu daného typu tokenu, který by byl zadán jako parametr funkce mint. Tato funkce by měla být snadno ovladatelná a uživatelsky přívětivá. Uživatel by měl být schopen snadno a rychle vytvořit nové NFT tokeny a přidat je do sbírky. Funkce by měla být navržena tak, aby umožňovala rychlé a snadné přidání nových NFT tokenů do sbírky a zároveň zaručila bezpečnost transakcí.

Možnost mintovat NFT pro whitelist uživatele

Tato funkce umožňuje vlastníkovi sbírky NFT tokenů vytvořit nové NFT tokeny pro uživatele na whitelistu. Tato funkce by měla být navržena tak, aby byla schopna ověřit, zda je uživatel na whitelistu a splňuje další požadované parametry. Pokud uživatel splňuje požadavky, měl by být schopen přidat nové NFT tokeny do své sbírky. Tato funkce by měla být bezpečná a zajišťovat, že pouze oprávnění uživatelé mají přístup k NFT tokenům na whitelistu. Vlastník sbírky by měl mít možnost kontrolovat proces mintování a schvalovat, kteří uživatelé mají přístup k novým NFT tokenům.

Možnost mintovat NFT

Tato funkce by měla být navržena tak, aby byla snadno ovladatelná a umožňovala vlastníkovi sbírky rychle a snadno přidávat nové NFT tokeny. Vlastník by měl mít možnost vybrat si, jaký druh NFT tokenu bude vytvořen. Funkce by měla být zabezpečena tak, aby nedocházelo k neoprávněným mintováním a mintování většího počtu kusů, než je povolený. Po dokončení procesu mintování by měl být nově vytvořený NFT token automaticky přidán do sbírky vlastníka. Celkově by tato funkce měla být navržena tak, aby byla bezpečná,

snadno ovladatelná a umožňovala vlastníkovi sbírky snadno a rychle přidávat nové NFT tokeny do své sbírky.

Umožnění nastavení flexibilního data na mintování

Tento požadavek umožňuje vlastníkovi sbírky NFT tokenů určit specifické datum, od kterého bude možné nové NFT tokeny mintovat. Tato funkce by měla být navržena tak, aby byla snadno ovladatelná a umožňovala vlastníkovi sbírky nastavit datum s jasným a přesným zobrazením. Vlastník by měl mít možnost nastavit flexibilní datum v budoucnosti, které bude určovat, kdy budou uživatelé schopni mintovat nové NFT tokeny. Toto nastavení by mělo být zabezpečeno tak, aby nedocházelo k neoprávněnému mintování před stanoveným datem a zároveň by umožňovalo vlastníkovi sbírky kontrolu nad procesem mintování.

Generování dynamických metadat

Tato funkcionalita umožňuje vlastníkovi sbírky NFT tokenů generovat dynamická metadata pro každý nově vytvořený NFT token. Tyto metadata mohou obsahovat různé informace, jako jsou například statistiky, vlastnosti a další charakteristiky NFT tokenu. Generování metadat by mělo být založeno na pseudonáhodných funkcích a mělo by být navrženo tak, aby umožňovalo vlastníkovi sbírky vytvářet unikátní NFT tokeny s různými vlastnostmi. Generování metadat by mělo být provedeno automaticky při vytváření nového NFT tokenu, což sníží čas potřebný k vytvoření a správě sbírky NFT tokenů. Celkově by tato funkce měla být navržena tak, aby umožňovala vlastníkovi sbírky NFT tokenů snadno a rychle vytvářet nové unikátní NFT tokeny s dynamickými metadaty, což přispěje k větší variabilitě a hodnotě celé sbírky.

Ukládání metadat na blockchain

Požadavek umožňuje vlastníkovi sbírky NFT tokenů ukládat metadata přímo na blockchain, aby byla zajištěna bezpečnost a integrita dat. Tato funkce by měla být navržena tak, aby umožňovala ukládání metadat do speciální proměnné na blockchainu, která by byla určena pouze pro tento účel. Vlastník by měl mít možnost vybrat si, která metadata budou ukládána na blockchain a v jakém formátu budou ukládána. Ukládání by mělo být prováděno přímo ze smart kontraktu pomocí funkce a mělo by být zabezpečeno tak, aby nedocházelo k neoprávněným úpravám dat. Ukládání metadat na blockchain zajišťuje, že informace o sbírce NFT tokenů jsou dostupné a chráněné před ztrátou dat nebo poruchami. Celkově by tato funkce měla být navržena tak, aby umožňovala snadné a rychlé ukládání metadat na

blockchain přímo ze smart kontraktu, což přispěje k větší bezpečnosti a důvěryhodnosti sbírky NFT tokenů

Možnost vybrat peníze z kontraktu

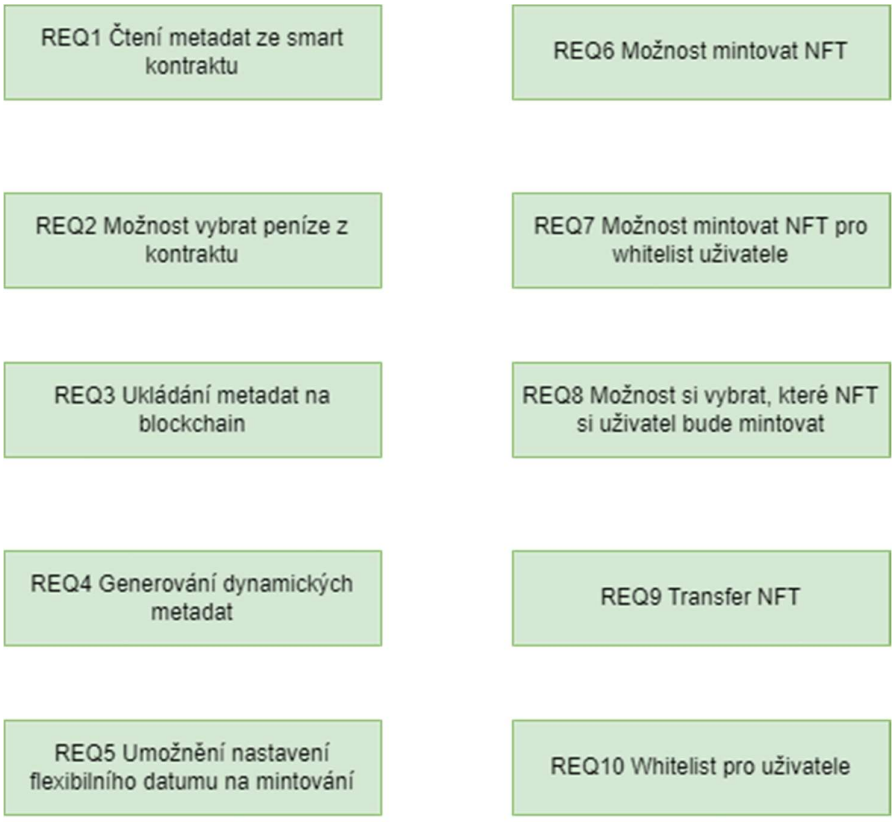
Funkcionalita umožňuje vlastníkovi sbírky NFT tokenů vybírat peníze z kontraktu. Tato funkce by měla být navržena tak, aby umožňovala vlastníkovi sbírky snadno a rychle vybírat peníze z kontraktu s maximální bezpečností. Vybírání peněz by mělo být prováděno pomocí funkce přímo ze smart kontraktu a mělo by být snadné k použití. Vlastník by měl mít možnost určit, kolik peněz chce vybrat a na jakou adresu chce peníze poslat. Tato funkce by měla být navržena tak, aby byla zabezpečena proti neoprávněnému vybírání peněz z kontraktu a aby byla snadno ověřitelná.

Čtení metadat ze smart kontraktu

Tento požadavek umožňuje uživatelům číst metadata ze smart kontraktu. Tato funkce by měla být navržena tak, aby umožňovala uživatelům snadno a rychle získat přístup k metadata sbírky NFT tokenů. Uživatelé a třetí strany by měli být schopni číst metadata z kontraktu pomocí speciální funkce v kontraktu a měli by být schopni číst metadata v formátu JSON. Funkce by měla být navržena tak, aby byla snadno použitelná a aby poskytovala uživatelům dostatečné množství informací o NFT tokenech.

6.1.1 Funkční požadavky

V aplikaci existuje deset různých funkčních požadavků, které pokrývají všechny funkcionality aplikace. Tyto požadavky lze sledovat na Obrázek 10.



Obrázek 10 Funkční požadavky smart kontrakt

REQ1 Čtení metadat ze smart kontraktu

Jedná se o funkční požadavek, který má za úkol poskytnout uživatelům a třetím stranám, možnost čtení onchain metadat NFT přímo z kontraktu. Na kontraktu se bude nacházet funkce, která po zavolání vrátí příslušná metadata k příslušenému tokenu.

REQ2 Možnost vybrat peníze z kontraktu

Tento funkční požadavek umožnuje vlastníkovy kontraktu vybrat specifikou sumu ETH z kontraktu na jeho adresu. Přímo na kontraktu se bude nacházet funkce, která právě toto umožní a po zavolání převede danou částku z kontraktu na peněženku.

REQ3 Ukládání metadat na blockchain

Funkční požadavek REQ3 Ukládání metadat na blockchain, má za úkol umožnit ukládat metadata přímo na chytrou smlouvu. Na kontraktu se nachází proměnná, do které se tyto metadata právě mapují na jednotlivé tokeny.

REQ4 Generování dynamických metadat

Jedná se o funkční požadavek, který má za úkol pseudonáhodně vygenerovat metadata pro proměnlivé atributy jako jsou právě HP, Gender, CombatPower. Na kontraktu se nachází funkce, která právě tyto pseudonáhodná metadata generuje a skládá z nich smysluplný json formát.

REQ5 Umožnění nastavení flexibilního datumu na mintování

Tento funkční požadavek, má za cíl umožnit vlastníkovy smart kontraktu, měnit a posouvat datum, od kdy bude zpřístupněný minting. Toho je docíleno pomocí funkce, která je vepsána do kontraktu. Tato funkce nastavuje jak začátek mintingu, tak i jeho konec.

REQ6 Možnost mintovat NFT

Funkční požadavek REQ6 Možnost mintovat NFT, má za cíl umožnit uživatelům mintovat NFT na jejich peněženku ve veřejném prodeji těchto tokenů. Toho je docíleno pomocí funkce, která určuje kolik tokenů, chce uživatel získat a jakého druhu, avšak maximální počet musí být také omezen. Tato funkce nejen umožní uživateli získat token, ale také provede veškeré transakce za daný token na smart kontrakt.

REQ7 Možnost mintovat NFT pro whitelist uživatele

Tento funkční požadavek má za cíl umožnit uživatelům mintovat NFT na jejich peněženku v privátním prodeji těchto tokenů (Whitelist). Toho je docíleno pomocí funkce, která určuje, jaký druh tokenu chce uživatel získat. Zároveň však tato funkce kontroluje, jestli se uživatel nachází na whitelistu. Tato funkce nejen umožní uživateli získat token, ale také provede veškeré transakce za daný token na smart kontrakt.

REQ8 Možnost si vybrat, které NFT si uživatel bude mintovat

Tento funkční požadavek má za úkol umožnit uživateli si vybrat NFT (jeden ze tří druhů). Toho je docíleno pomocí přidání parametrů do všech potřebných funkcí.

REQ9 Transfer NFT

Jedná se o funkční požadavek, který umožnuje transfer NFT mezi jednotlivými uživateli. Toho je docíleno pomocí funkcí, které implementují všechny standardy.

REQ10 Whitelist pro uživatele

Funkční požadavek REQ10 Whitelist, se zaměřuje na vytvoření merkle tree přístupu k privátnímu prodeji. Toho je docíleno tak, že je vytvořena proměnná, do které se ukládají právě podepsané adresy.

6.1.2 Nefunkční požadavky

V aplikaci se nachází tři nefunkční požadavky. Všechny tyto nefunkční požadavky popisují systém. Lze je vidět na Obrázek 11.

Obrázek 11 Nefunkční požadavky smart kontrakt

NREQ1 Bezpečnost

U tohoto požadavku je kladen důraz na zabezpečení aplikace při použití smart kontraktu je kód po deployi již nemění a veřejný, tudíž je nutno brát v potaz, všechny možné zabezpečení a otestovat celý kontrakt na testnetu.

NREQ2 Škálovatelnost

Tento nefunkční požadavek popisuje škálovatelnost je to klíčový požadavek pro blockchainové aplikace, protože tyto aplikace se obvykle používají pro velké množství uživatelů a transakcí. Škálovatelnost zahrnuje schopnost systému zvládat rostoucí zátěž, aby mohl stále poskytovat rychlou a spolehlivou službu bez výrazného poklesu výkonu. Toho je docíleno vhodnou volbou blockchainu, na který se tento smart kontrakt nasadí

NREQ3 Výkon

Nefunkční požadavek NREQ3 Výkon je dalším důležitým požadavkem pro blockchainové aplikace, protože tyto aplikace musí být schopny rychle a efektivně zpracovávat transakce a poskytovat rychlou odezvu uživatelům. Výkon zahrnuje rychlost zpracování transakcí, reakční čas aplikace, odezvu uživatelského rozhraní a celkovou efektivitu systému. Výkon je ovlivněn nejen kvalitou kódu, ale hlavně výběrem blockchainu.

6.2 Výběr vhodného standardu

Vybrání vhodného standardu je velmi důležité, protože, jak již bylo zmíněno, při ostrém provozu již kód nelze měnit bez toho, aniž by uživatelé ztratili všechny původní tokeny. Z tohoto důvodu bylo kladen důraz na to, aby standard splňoval všechny moderní požadavky a zároveň všechny funkční a nefunkční požadavky, které smart kontrakt vyžaduje. Proto byl zvolen typický standard ERC-721, tj. NFT standard, který je možné pozorovat prakticky u všech NFT kolekcí. Byly zváženy standardy ERC-721 od OpenZeppelin a ERC-721a od Chiru labs, který rozšiřuje základní standard. Nakonec byl zvolen klasický ERC-721 s implementací od OpenZeppelin, konkrétně s rozšířením Enumerable, což nám dovolí iterovat přes všechny tokeny na kontraktu, což je důležité pro zobrazování všech NFT a pro náš účel. Velmi důležitou součástí je také rozšíření tohoto standardu o třídu Ownable, abychom zajistili, že určité funkce bude moci volat pouze adresa, která kontrakt vlastní, což zvýší bezpečnost kontraktu. Tabulka 2 ukazuje funkce, které se nachází na ERC721Enumerable.

Tabulka 2 ERC721Enumerable základní funkce

ERC721Enumerable			
supportsInterface(interfaceId)	tokenOfOwnerByIndex(owner, index)	totalSupply()	tokenByIndex(index)
_beforeTokenTransfer(from, to, firstTokenId, batchSize)	balanceOf(owner)	ownerOf(tokenId)	name()
symbol()	tokenURI(tokenId)	_baseURI()	approve(to, tokenId)
getApproved(tokenId)	setApprovalForAll(operator, approved)	isApprovedForAll(owner, operator)	transferFrom(from, to, tokenId)
safeTransferFrom(from, to, tokenId)	safeTransferFrom(from, to, tokenId, data)	_safeTransfer(from, to, tokenId, data)	_ownerOf(tokenId)
_exists(tokenId)	_isApprovedOrOwner(spender, tokenId)	_safeMint(to, tokenId)	_safeMint(to, tokenId, data)
_mint(to, tokenId)	_burn(tokenId)	_transfer(from, to, tokenId)	_approve(to, tokenId)
_setApprovalForAll(owner, operator, approved)	_requireMinted(tokenId)	_afterTokenTransfer(from, to, firstTokenId, batchSize)	__unsafe_increaseBalance(account, amount)

6.3 Ukládání metadata (databáze pro NFT)

Ukládání metadat je nezbytnou součástí každé NFT kolekce, protože metadata určují formát této kolekce, její vlastnosti a vzhled. Tyto metadata se nejčastěji ukládají off-chain, protože se tyto kolekce nacházejí na Ethereum, kde poplatky za zápis na chain nejsou úplně nízké.

To však vede k mnoha problémům, jako je centralizace, protože je běžné ukládat tato data na IPFS a jiné podobné služby třetích stran, kde je třeba platit za uložená data a v případě poruch nejsou tato data k dispozici. Jako jedno z řešení se nabízí ukládání dat přímo na blockchain, což je v případě Etherea velmi drahé kvůli gas fees. V našem případě je to však ideální řešení, protože plánujeme použít blockchain, který je primárně určen pro hry, jako například SKALE. Díky tomu se vyhneme poplatkům třetím stranám a nemusíme vytvářet vlastní centralizované úložiště s robustním backendem a bezpečnou API. Data budou trvale zapsána na blockchainu, což nám zajistí větší důvěryhodnost hráčů v tyto tokeny. Proto jsme se rozhodli nevyužít klasickou databázi pro ukládání těchto tokenů, ale volili jsme databázi, která je vytvořena přímo na blockchainu. Toho jsme dosáhli pomocí proměnné "mapping(uint256 => string) private _metadata", která uchovává jednotlivé informace pomocí mapování k jednotlivým tokenům podle jejich ID.

6.3.1 Struktura Metadata

Struktura metadata, je zvolena dle požadavků aplikace, tj jsou vytvořené tři základní NFT tokeny, u kterých se vždy liší obrázek, název, typ a popis. Tyto věci se mění vždy na základě zvoleného monstra a to bud Komodomon, Hippomon a nebo Girrafon. Dále se pak pro každý token vygenerují náhodné vlastnosti jako je gender, height, weight a combat power. Tyto data se generují pseudonáhodně a slouží pouze pro testování, při ostrém provozu bude nutno využít opravdových generátorů třetích stran jako jsou například oracles, protože vygenerovat „náhodné“ číslo na blockchainu je velmi složitá disciplína.

Data jsou zapsána na chain v následující Json struktuře, kde pro každý token se generují unikátní hodnoty.

{

***"name"**: "Girrafon",*

***"description"**: "Girrafon is a majestic creature that roams the open plains of the world. Its body is tall and slender, with long, spindly legs that seem to stretch on forever. Its coat is a deep reddish-orange, and its mane and tail blaze with flames that never seem to go out. As a Fire-type monster, Girrafon is incredibly powerful, with the ability to summon and control flames with ease. Its long neck can reach up to catch even the highest of branches and its hooves can ignite the ground it walks on, making it a fearsome opponent in battle. Despite its ferocious appearance, Girrafon is a gentle and peaceful creature at heart. It spends its*

days grazing on the sweet grasses of the plains and basking in the warmth of the sun. It is fiercely loyal to its trainers and will stop at nothing to protect them from harm.",

*"**image**":"https://cdn.midjourney.com/b61da443-e0dd-4e77-bbf4-06f7b425e314/grid_0.png",*

*"**edition**": 1,*

*"**weight**": 47,*

*"**height**": 58,*

*"**type**": "Grass",*

*"**gender**": "male",*

*"**CombatPower**": 75*

}

6.4 Implementace

Tato část je zaměřena na popis implementace jednotlivých funkcí a ukazuje základní přehled fungování aplikace. Pro implementaci byl zvolen standard ERC721 s rozšířením Enumerable a integrací třídy Ownable, která zajišťuje bezpečnost volání funkcí. Využívá také kryptografickou knihovnu MerkleProof, která zajišťuje právě whitelist. Dále je také využita implementace Stringu od společnosti OpenZeppelin. Celý kód je napsán v jazyku Solidity, a to s podporou verze 0.8.9 a vyšší.

setListRoot

Tato funkce nazvaná "setListRoot" slouží k nastavení hodnoty proměnné "ListRoot". Funkce je označena jako "external" a díky tomu, může být volána z jiných kontraktů. Přívlastek "onlyOwner" ve funkci zajišťuje, že pouze vlastník kontraktu má oprávnění ke změně hodnoty proměnné "ListRoot". Proměnná "ListRoot" je typu "bytes32", takže při nastavování hodnoty je nutné provést přetypování vstupního argumentu "root" na tento typ.

randomBetween

Funkce nazvaná "randomBetween" slouží k generování náhodného čísla v určitém intervalu. Funkce je označena jako "private" a "view", což znamená, že může být volána pouze zevnitř kontraktu a neprovádí žádné změny stavu blockchainu. Vstupními argumenty funkce jsou:

"tokenId": identifikátor tokenu, který má být použit k generování náhodného čísla.

"min": dolní mez, v rámci, které se má generovat náhodné číslo.

"max": horní mez, v rámci, které se má generovat náhodné číslo.

A vrací hodnotu typu "uint256".

Výpočet náhodného čísla se provádí pomocí funkce "keccak256", která vrací hash hodnotu z concatenace několika hodnot: časového údaje aktuálního bloku "block.timestamp", náhodné hodnoty předchozího bloku "block.prevrandao" a identifikátoru tokenu "tokenId". Tyto hodnoty jsou nejprve zakódovány pomocí funkce "abi.encodePacked" a následně zahashovány pomocí funkce "keccak256".

Výsledný hash se pak použije k výpočtu náhodného čísla tak, že se výsledek vynásobí modulo (operátor "%") rozdílem mezi horní a dolní mezí a k výsledku se přičte dolní meze. Funkce nakonec vrátí výsledné náhodné číslo.

generateRandomMetadata

Tato funkce v Solidity nazvaná "generateRandomMetadata" slouží k generování náhodných metadat pro token. Funkce je označena jako "private" a "view", což znamená, že může být volána pouze zevnitř kontraktu a neprovádí žádné změny stavu blockchainu.

Vstupními argumenty funkce jsou:

"tokenId": identifikátor tokenu, který má být použit k generování metadat.

"tokenName": jméno tokenu, které bude použito k výpočtu metadat.

Funkce vrací řetězec typu "string memory".

Funkce nejprve volá funkci "randomBetween" čtyřikrát s různými identifikátory tokenu a intervaly, aby vytvořila náhodné hodnoty pro weight, height, combat power a gender tokenu. Tyto hodnoty se poté použijí k výpočtu jednotlivých částí metadat pomocí funkcí "tokenNameToString", "tokenNameToDescription", "tokenNameToType" a "tokenNameToURL".

Výsledné metadat jsou poté složeny do jednoho řetězce pomocí funkce "abi.encodePacked" a funkce nakonec vrací výsledný řetězec metadat.

tokenNameToDescription

Funkce "tokenNameToDescription" slouží k výpočtu popisu pro token na základě jeho jména. Funkce je označena jako "private" a "pure", což znamená, že může být volána pouze zevnitř kontraktu a neprovádí žádné změny stavu blockchainu, ani nevolá žádné jiné funkce.

Vstupním argumentem funkce je "tokenName", který reprezentuje jméno tokenu a vrací řetězec typu "string memory".

Funkce obsahuje řadu podmínek, které používají operátor "==" k porovnání jména tokenu s jednotlivými možnými hodnotami typu "TokenName". V případě, že jméno tokenu odpovídá jedné z možných hodnot, funkce vrátí popis příslušného tokenu jako řetězec. V opačném případě funkce vrátí řetězec s popisem jiného tokenu.

tokenNameToString

Funkce nazvaná "tokenNameToString" slouží k výpočtu jména tokenu na základě jeho jména (vstupního parametru). Funkce je označena jako "private" a "pure", což znamená, že může být volána pouze zevnitř kontraktu a neprovádí žádné změny stavu blockchainu, ani nevolá žádné jiné funkce.

Vstupním argumentem funkce je "tokenName", který reprezentuje jméno tokenu a vrací řetězec typu "string memory".

Funkce obsahuje řadu podmínek, které používají operátor "==" k porovnání jména tokenu s jednotlivými možnými hodnotami typu "TokenName". V případě, že jméno tokenu odpovídá jedné z možných hodnot, funkce vrátí jméno příslušného tokenu jako řetězec. V opačném případě funkce vrátí jméno jiného tokenu.

tokenNameToType

Tato funkce v Solidity nazvaná "tokenNameToType" slouží k výpočtu typu tokenu na základě jeho jména. Funkce je označena jako "private" a "pure", což znamená, že může být volána pouze zevnitř kontraktu a neprovádí žádné změny stavu blockchainu, ani nevolá žádné jiné funkce.

Vstupním argumentem funkce je "tokenName", který reprezentuje jméno tokenu a vrací řetězec typu "string memory"

Funkce obsahuje řadu podmínek, které používají operátor "==" k porovnání jména tokenu s jednotlivými možnými hodnotami typu "TokenName". V případě, že jméno tokenu

odpovídá jedné z možných hodnot, funkce vrátí typ příslušného tokenu jako řetězec. V opačném případě funkce vrátí typ jiného tokenu.

tokenNameToURL

Tato funkce v Solidity, nazvaná "tokenNameToURL", slouží ke konverzi jména tokenu na odpovídající URL obrázku. Funkce je označena jako "private" a "pure", což znamená, že může být volána pouze uvnitř kontraktu a neprovádí žádné změny stavu blockchainu ani nevolá žádné jiné funkce.

Vstupním argumentem funkce je "tokenName", který reprezentuje jméno tokenu, a výstupem je řetězec typu "string memory", který obsahuje URL obrázku.

Funkce obsahuje řadu podmínek, které používají operátor "==" k porovnání jména tokenu s jednotlivými možnými hodnotami typu "TokenName". V případě, že jméno tokenu odpovídá jedné z možných hodnot, funkce vrátí URL obrázku příslušného tokenu. V opačném případě funkce vrátí URL jiného tokenu.

setTokenMetadata

Funkce nazvaná "setTokenMetadata" slouží k nastavení metadat konkrétního tokenu. Funkce je označena jako "internal" a "onlyOwner", což znamená, že může být volána pouze uvnitř kontraktu a pouze vlastníkem kontraktu.

Vstupními argumenty funkce jsou "tokenId", který reprezentuje ID tokenu, a "metadata", které představují metadaty tokenu jako řetězec typu "string memory".

Funkce obsahuje volání interní funkce "_requireMinted", která ověřuje, že token s daným ID byl již minted. Pokud byl token minted, hodnota "metadata" je uložena do mapy "_metadata" s klíčem "tokenId".

getTokenMetadata

Funkce "getTokenMetadata" slouží k získání metadat konkrétního tokenu. Funkce je označena jako "external" a "view", což znamená, že může být volána ze všech směrů a neprovádí žádné změny stavu blockchainu.

Vstupním argumentem funkce je "tokenId", který reprezentuje ID tokenu. Funkce vrací řetězec typu "string memory", který představuje metadaty tokenu.

Funkce obsahuje volání interní funkce "_requireMinted", která ověřuje, že token s daným ID byl již minted. Pokud byl token minted, metadata tokenu jsou získány z mapy "_metadata" pomocí klíče "tokenId" a následně jsou vráceny jako výstup funkce.

presaleMint,

Funkce nazvaná "presaleMint", slouží k mintování tokenů v rámci předprodeje. Funkce je označena jako "external" a "payable", což znamená, že může být volána ze všech směrů a může přijímat platby.

Vstupními argumenty funkce jsou "ListProof", který slouží jako důkaz k prokázání se v rámci předprodeje, a "monsterType", který představuje typ monster, které chce uživatel získat.

Funkce obsahuje několik podmínek "require", které slouží k ověření vstupů. Tyto kontroly zahrnují ověření, že typ monstra je platný, že je aktivní předprodej, že uživatel ještě nemintoval, že uživatel je na whitelist, že uživatel zaplatil správnou cenu a že listproof je validní a není povdrhnutý.

Pokud všechny kontroly projdou, funkce nastaví hodnotu pro uživatele v mapě "WhitelistMinted" na "true" čímž ho označní, že již mintoval a zvýší počet tokenů v předprodeji a zavolá funkci "_safeMint". Následně se volá funkce "generateRandomMetadata" k vytvoření metadat pro právě mintnutý token a pak funkci "setTokenMetadata" k nastavení těchto metadat.

publicSaleMint

Tato funkce v Solidity, nazvaná "publicSaleMint", slouží k minting tokenů v rámci veřejného prodeje. Funkce je označena jako "external" a "payable", což znamená, že může být volána ze všech směrů a může přijímat platby.

Vstupními argumenty funkce jsou "quantity", která představuje počet minted tokenů, a "monsterType", který představuje typ monstera.

Funkce obsahuje několik volání funkce "require", které slouží k ověření podmínek. Tyto kontroly zahrnují ověření, že typ monster je platný, že je aktivní veřejný prodej, že není vyprodáno a že uživatel již nemintoval maximální počet tokenů během veřejného prodeje. Funkce také ověřuje, že uživatel zaplatil správnou částku.

Pokud všechny kontroly projdou, funkce zvýší počet tokenů mintnutý v mapě "MintedByAddress" a zavolá funkci "_safeMint" pro každý z minted tokenů. Následně funkce volá funkci "generateRandomMetadata" k vytvoření metadat pro každý z minted tokenů a pak funkci "setTokenMetadata" k nastavení těchto metadat.

setPresale

Funkce nazvaná "setPresale", slouží k nastavení časových intervalů pro předprodej tokenů. Funkce je označena jako "external" a "onlyOwner", což znamená, že může být volána ze všech směrů pouze vlastníkem kontraktu.

Vstupními argumenty funkce jsou "start" a "end", které představují začátek a konec časového intervalu pro předprodej, tyto udaje se zadávají v unix timestampu.

Funkce nastavuje hodnoty "ListStartTime" a "ListEndTime" na hodnoty vstupních argumentů "start" a "end". Tyto hodnoty používá funkce "presaleMint" k ověření, zda je předprodej aktivní.

tokenURI

Tato funkce v Solidity, nazvaná "tokenURI", slouží k vrácení identifikátoru pro zadaný token. Funkce je označena jako "public", "view" a "virtual override", což znamená, že může být volána ze všech směrů bez nutnosti platby a že jde o virtuální funkci, kterou lze přepsat v odvozených třídách.

Vstupním argumentem funkce je "tokenId", který představuje identifikátor tokenu. Funkce vrací řetězec typu "string memory".

Funkce volá funkci "_requireMinted" k ověření, zda byl token s daným identifikátorem minted. Následně funkce použije identifikátor tokenu k vyhledání metadat tokenu v mapě "_metadata". Pokud metadat nejsou prázdná, funkce vrátí metadat jako řetězec. V opačném případě funkce vrátí prázdný řetězec.

Withdraw

Funkce nazvaná "withdraw", slouží k výběru peněz z kontraktu na vlastní peněženku. Funkce je označena jako "external" a "onlyOwner", což znamená, že může být volána ze všech směrů pouze vlastníkem kontraktu.

Funkce získá zůstatek kontraktu pomocí volání "address(this).balance" a následně volá funkci "payable(msg.sender).transfer" k převodu zůstatku na peněženku vlastníka. Tím se zajistí, že vlastník kontraktu může vybrat jakýkoli zůstatek, který se nachází v kontraktu.

6.5 Deploy

Deploy kontraktu byl proveden na testnet Mumbai a lze jej najít pod adresou 0x349be5b36718f1f550AD1794Cf729aF35A448C92.

Tento deploy probíhal skrze hardhat a script, který vezme data z konfiguračního souboru a vložít je do kódu, který lze vidět na Obrázek 12.

```
const hre = require("hardhat");

async function main() {
    const [deployer] = await hre.ethers.getSigners();
    console.log("Deploying contract with account:", deployer.address);

    const Monsters = await hre.ethers.getContractFactory("Monsters");
    const monsters = await Monsters.deploy();

    await monsters.deployed();

    console.log("Monsters contract deployed at:", monsters.address);
}

main()
    .then(() => process.exit(0))
    .catch((error) => {
        console.error(error);
        process.exit(1);
    });
```

Obrázek 12 Deploy Script

Konfigurační soubor, slouží k uložení potřebných dat, jako je například INFURA_API_KEY, který slouží jako RPC node anebo MNEMONIC, který slouží jako privátní klíč k deploy peněžence. Tento konfigurační soubor lze vidět na Obrázek 13.

```
require("@nomiclabs/hardhat-ethers");

const { MNEMONIC, INFURA_API_KEY } = process.env;

module.exports = {
    solidity: "0.8.9",
    networks: {
        mumbai: {
            url: `https://polygon-mumbai.infura.io/v3/${INFURA_API_KEY}`,
            accounts: {
                mnemonic: MNEMONIC,
            },
        },
    },
};
```

Obrázek 13 Konfigurační soubor

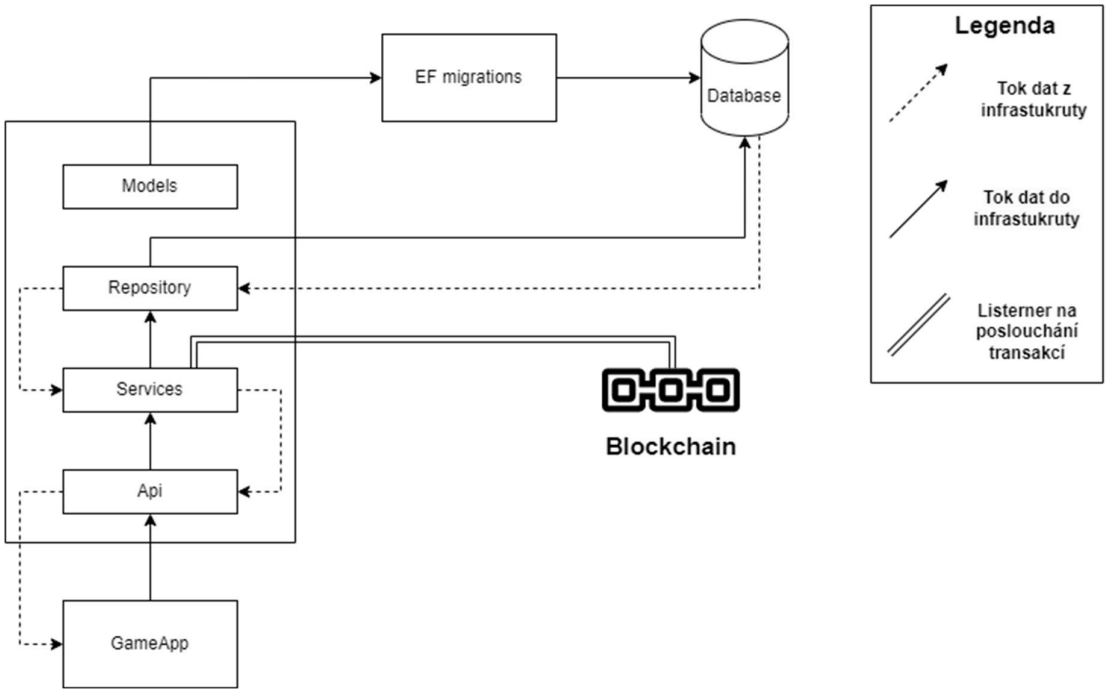
Všechno tyto scripty lze následně spustit pomocí konzolového příkazu „npx hardhat run --network mumbai scripts/deploy.js“ po jeho spuštění, je do konzole zobrazena adresa, na které lze najít finální kolekci.

7 NÁVRH A REALIZACE DATABÁZE A BACKENDU

Tato kapitola se zaměřuje na návrh a realizaci databáze, která má sloužit pro ukládání informací o uživatelích a blockchainových transakcích. Databáze je navržena jako PostgreSQL a obsahuje momentálně pouze dvě tabulky pro ukládání uživatelů a informací o blockchainových transakcích. Tento návrh je vytvořen s důrazem na jednoduchost a udržitelnost. Databáze je úzce propojena s backendovou infrastrukturou, která je postavena na čtyřvrstvé architektuře, napsané v jazyce C# s využitím frameworku Dotnet. V této aplikaci se používá přístup pomocí repository patternu a migračního frameworku Entity k databázi, což zajišťuje robustní strukturu a zabezpečení.

Celá infrastruktura aplikace se skládá z modelů, které určují strukturu databáze a tabulek, repository pro přístup k datům, services pro složitější datové operace a API pro vystavení endpointů pro připojení aplikací. Tento design nejen určuje strukturu modelů a tabulek, ale také zajišťuje robustnost a bezpečnost přístupu k databázi.

Aplikace a databáze zatím není nikde hostovaná a slouží pouze jako testovací prostředí. V této části je vytvořena pouze implementace databáze, modelů a repository přístupu. Diagram workflow aplikace lze vidět na obrázku Obrázek 14.



Obrázek 14 Workflow aplikace

7.1 Použité technologie

Tato část kapitoly se zaměřuje na popis použitých technologií pro návrh a realizaci databáze a backendové infrastruktury. Mezi hlavní použité technologie patří PostgreSQL, což je open source databázový systém, zvolený pro ukládání dat díky podpoře vysoké úrovně SQL standardů a možnosti implementace na všechny velké cloudové platformy, jako jsou AWS nebo Google Cloud. Další využitou technologií je jazyk C# a framework Dotnet, který umožňuje rychlou a bezpečnou tvorbu backendové infrastruktury.

Pro přístup k databázi je využit repository pattern, který odděluje přístup k databázi od práce s daty a API, čímž zajišťuje robustnost a bezpečnost aplikace. Migrační framework Entity zde slouží k řízení změn v databázi a k vytváření její struktury. Tímto způsobem lze jednoduše řešit změny v databázi a umožnit více vývojářům pracovat na jednom projektu bez narušení celistvosti práce.

7.2 Databázové modely

Tato část je velmi důležitá, protože nám nejen formuje budoucí vývoj, ale určuje celou strukturu databáze a tabulek.

7.2.1 GameAccount

Nachází se zde třída GameAccount, která slouží k vytvoření tabulky na ukládání herních účtů a dat. Tato třída obsahuje následující atributy, které je možno vidět v Tabulka 3. V tabulce je Id použito jako primární klíč a je generováno pomocí funkce Guid, která vrátí unikátní identifikátor. Dále jsou zde také vytvořené dva indexy, a to pro rychlé vyhledávání v datech, jedná se o indexy na sloupeček Address a ChainKind, index na primární klíč je vytvořen automaticky.

Tabulka 3 GameAccount

Název	Datový typ	Popis
Id	GUID	Unikátní identifikátor herního účtu
Address	Text	Adresa herního účtu jako textový řetězec, může být null
ChainKind	Enum	Typ blockchain sítě, na které je herní účet registrován
Email	Text	E-mailová adresa přiřazená k hernímu účtu, může být null
CreatedAt	DateTime	Datum a čas vytvoření herního účtu, v UTC časovém pásmu

UpdatedAt	DateTime	Datum a čas poslední aktualizace herního účtu, v UTC časovém pásmu
DeletedAt	DateTime	Volitelné datum a čas, kdy byl herní účet smazán

7.2.2 BlockchainTransaction

Dále se zde nachází třída BlockchainTransaction, která slouží k vytvoření tabulky pro ukládání informací o provedených transakcích. Ve třídě se nachází atributy, které lze pozorovat Tabulka 4. V této tabulce je jako primární klíč využito Id, které se generuje skrze funkcí Guid, která vrací unikátní identifikátor. Nachází se zde také UserId, který zde slouží jako primární klíč, aby se dalo filtrovat, jaký uživatel udělal, jakou transakci. Najdeme zde také čtyři databázové indexy, které zde slouží pro rychlé vyhledávání, tyto indexy jsou na sloupce UserId, Hash, ChainKind a Suceeded.

Tabulka 4 BlockchainTransaction

Název	Datový typ	Popis
Id	GUID	Unikátní identifikátor transakce
UserId	GUID	Unikátní identifikátor uživatele, který provedl transakci
Hash	Text	Hash transakce
ChainKind	Enum	Typ blockchain sítě, na které byla transakce provedena
Succeeded	Boolean	Informace o úspěšnosti provedení transakce
AttemptedAt	DateTimeOffset	Datum a čas pokusu o provedení transakce, v UTC časovém pásmu
CreatedAt	DateTimeOffset	Datum a čas vytvoření záznamu o transakci, v UTC časovém pásmu
UpdatedAt	DateTimeOffset	Datum a čas poslední aktualizace záznamu o transakci, v UTC časovém pásmu
DeletedAt	DateTimeOffset	Volitelné datum a čas, kdy byla transakce smazána

7.2.3 MyDBContext

Tyto modely, však představují pouze objekty a je nutno, aby byly vytvořené a vložené přímo do databáze. Od toho nám zde slouží třída MyDBContext, která právě toto zajišťuje. V této třídě definujeme DbSet vlastnosti, které právě představují naše tabulky v databázi, a umožnují nám s nimi manipulovat v aplikaci.

public DbSet<GameAccount> game_accounts { get; set; } = null!;

public DbSet<BlockchainTransaction> blockchain_transactions { get; set; } = null!;

Velmi důležitou součástí této třídy je také definice našeho vlastního datového typu ChainKind, který je nutno vytvořit pomocí NpsqlConnection.GlobalTypemapper, který právě slouží k vytváření vlastních datových typů pro PostgreSQL. Dále se zde také nachází model builder, který slouží k definování struktury databáze pomocí modelů právě pomocí Entity Framework Core.

Pro entitu GameAccount byla definována vlastnost Id jako primární klíč pomocí metody HasKey. Dále byla nastavena maximální délka vlastnosti Address na 42znaků pomocí metody HasMaxLength. Vlastnosti ChainKind, CreatedAt a UpdatedAt byly nastaveny jako povinné pomocí metody IsRequired. Vlastnost Email byla nastavena na maximální délku 255, ale může být i null

Pro entitu BlockchainTransaction byla opět definována vlastnost Id jako primární klíč. Byl také vytvořen index pro vlastnost UserId pomocí metody HasIndex. Vlastnost Hash byla omezena na maximální délku 64. Vlastnosti ChainKind a Succeeded byly nastaveny jako povinné. Vlastnost AttemptedAt může být null. Podobně jako u GameAccount byly nastaveny vlastnosti CreatedAt a UpdatedAt jako povinné. Vlastnost DeletedAt může být null.

7.3 Repository vzor

Repository vzor byl vybrán hlavně z důvodu, že nám zde vytváří jednotné rozhraní mezi aplikační vrstvou a databází. Díky tomuto přístupu, je poté přístup přímo k databázi skryt zvenčí a tím, je zvýšená bezpečnost. Velkou výhodou je také možnost zjednodušení testování a zajištění konzistentního přístupu k datům. Tento přístup, je vždy oddělen do jednotlivých tříd, dle přístupu k jednotlivým tabulkám.

7.3.1 GameAccountRepository

Nejprve se zde nachází třída GameAccountRepository, která dědí od třídy IGameAccountRepository v tomto interface se nachází definice hlaviček všech příslušných funkcí, které nám zase usnadňují a zpřehledňují vývoj.

GetOrCreate

Funkce nejprve hledá v databázi účet s danou adresou, typem blockchainu (ChainKind) a nulovým časem smazání (DeletedAt), toto provádí pomocí metody FirstOrDefault a předáních podmínek v lambda výrazu. Pokud účet neexistuje, vytváří novou instanci třídy

GameAccount s adresou a typem blockchainu. Tento nový účet je pak přidán do kolekce game_accounts v databázi a změny jsou uloženy pomocí metody SaveChanges. Pokud však neexistuje vytvoří nový účet dle zadaných parametrů. Funkce tedy vrátí buď nově vytvořený účet, nebo nalezený účet z databáze.

GetByAddress

Funkce nejprve hledá v databázi herní účet s danou adresou a typem blockchainu a nulovým časem smazání (DeletedAt). Toto provádí pomocí metody SingleOrDefault a předaných podmínek v lambda výrazu. Pokud účet s danou adresou a typem blockchainu neexistuje v databázi, funkce vrátí hodnotu null. Pokud účet s danou adresou a typem blockchainu existuje v databázi, funkce vrátí nalezený účet jako objekt typu GameAccount.

Insert

Tato funkce Insert slouží k vložení nového herního účtu (GameAccount) do databáze. Funkce nejprve přidá nový herní účet do kolekce game_accounts v databázi, a to pomocí metody Add a předaného objektu herního účtu. Následně funkce uloží změny v databázi pomocí metody SaveChangesAsync a po dokončení operace vrátí hodnotu, jestli se daná operace povedla. Důležitou součástí této funkce je také její asynchronní chování, což nám dovolí škálovat naší aplikaci.

UpdateEmail

Funkce pracuje s databázovým dotazem pomocí metody ExecuteSqlRawAsync, která přijímá SQL dotaz a parametry dotazu. Funkce vytváří dotaz SQL s parametry pro aktualizaci e-mailové adresy a data a času poslední aktualizace v tabulce game_accounts pro řádek s konkrétním Id.

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset.

Nakonec funkce provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync a předá mu parametry pro výraz @Email, @UpdatedAt a @Id jako instance třídy SqlParameter. Tento dotaz provede aktualizaci e-mailové adresy a data a času poslední aktualizace pro řádek s konkrétním Id v databázi.

Update

Tato funkce Update slouží k aktualizaci existujícího herního účtu (GameAccount) v databázi.

Funkce používá přímé SQL dotazy a výrazy @Address, @ChainKind, @Email, @UpdatedAt a @Id jsou použity jako parametry v dotazu.

Funkce nejprve vytvoří dotaz SQL s parametry. Tento dotaz aktualizuje sloupce Address, ChainKind, Email a UpdatedAt v tabulce game_accounts pro řádek s konkrétním Id.

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset.

Nakonec funkce provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync a předá mu parametry pro výrazy @Address, @ChainKind, @Email, @UpdatedAt a @Id jako instance třídy SqlParameter. Tento dotaz provede aktualizaci údajů o herním účtu pro řádek s konkrétním Id v databázi.

Delete

Tato funkce Delete slouží k označení herního účtu (GameAccount) jako smazaného v databázi.

Funkce používá přímé SQL dotazy a výrazy @DeletedAt a @Id jsou použity jako parametry v dotazu.

Funkce nejprve vytvoří dotaz SQL s parametry. Tento dotaz nastaví hodnotu sloupce DeletedAt na aktuální datum a čas v tabulce game_accounts pro řádek s konkrétním Id.

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset.

Nakonec funkce provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync a předá mu parametry pro výrazy @DeletedAt a @Id jako instance třídy SqlParameter. Tento dotaz provede označení herního účtu jako smazaného pro řádek s konkrétním Id v databázi.

GetAll

Tato funkce GetAll slouží k získání seznamu všech herních účtů (GameAccount) z databáze.

Funkce nejprve používá metodu Where k nalezení všech herních účtů, u kterých sloupec DeletedAt obsahuje hodnotu null (tj. účet nebyl smazán). Tato metoda vrací výsledek jako IQueryable. Následně převede tuto hodnotu pomocí metody ToList a vrátí seznam všech herních učtů, které nebyly v databázi označené jako smazané

GetById

Tato funkce GetById slouží k získání konkrétního herního účtu (GameAccount) z databáze pomocí jeho Id.

Funkce nejprve používá metodu FindAsync k nalezení herního účtu podle zadaného Id a výsledek uloží do proměnné gameAccount. Funkce používá metodu Result, aby získala výsledek asynchronního dotazu a počkala na jeho dokončení.

Následně funkce kontroluje, zda výsledek není null a zda sloupec nebyl smazán. Pokud tomu tak není, funkce vrátí null v opačném případě vrátí nalezený účet.

7.3.2 BlockchainTransactionRepository

Dále zde také najdeme třídu BlockchainTransactionRepository, která dědí třídu IBlockchainTransactionRepository. Popis veškerých funkcí najdeme níže.

Insert

Tato funkce Insert slouží k vložení nové transakce blockchainu (BlockchainTransaction) do databáze.

Funkce nejprve přidá novou transakci do kolekce blockchain_transactions v databázi, a to pomocí metody Add a předaného objektu transakce. Následně funkce uloží změny v databázi pomocí metody SaveChangesAsync a po dokončení operace vrátí hodnotu, jestli se daná operace povedla.

InsertRange

Tato funkce InsertRange slouží k vložení více transakcí v blockchainu (BlockchainTransaction) do databáze najednou.

Funkce nejprve přidá nové transakce do kolekce blockchain_transactions v databázi pomocí metody AddRange a předaného seznamu objektů transakcí. Použitím metody Set a předáním typu BlockchainTransaction získá funkce kolekci objektů této třídy.

Následně funkce uloží změny v databázi pomocí metody SaveChangesAsync a po dokončení operace vrátí hodnotu, jestli se daná operace povedla.

MarkSucceeded

Tato funkce MarkSucceeded slouží k označení transakce v blockchainu (BlockchainTransaction) jako úspěšné nebo neúspěšné v databázi.

Funkce používá přímé SQL dotazy a výrazy @Succeeded, @UpdatedAt, @AttemptedAt a @Id jsou použity jako parametry v dotazu.

Funkce nejprve vytvoří dotaz SQL s parametry. Tento dotaz nastaví hodnotu sloupce Succeeded na zadanou hodnotu (true nebo false), aktualizuje hodnotu sloupce UpdatedAt na aktuální datum a čas a nastaví hodnotu sloupce AttemptedAt na aktuální datum a čas v tabulce BlockchainTransactions pro řádek s konkrétním Id, pokud nebyl záznam smazán.

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset a provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync.

Delete

Tato funkce Delete slouží k označení transakce v blockchainu (BlockchainTransaction) jako smazané v databázi.

Funkce používá přímé SQL dotazy a výrazy @DeletedAt a @Id jsou použity jako parametry v dotazu.

Funkce nejprve vytvoří dotaz SQL s parametry. Tento dotaz nastaví hodnotu sloupce DeletedAt na aktuální datum a čas v tabulce BlockchainTransactions pro řádek s konkrétním Id, pokud řádek ještě nebyl smazán.

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset.

Nakonec funkce provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync a předá mu parametry. Tento dotaz provede označení transakce v blockchainu jako smazané pro řádek s konkrétním Id v databázi.

GetByHash

Tato funkce GetByHash slouží k získání transakce v blockchainu (BlockchainTransaction) z databáze pomocí jejího hashového identifikátoru a typu blockchainu (ChainKind).

Funkce používá metodu SingleOrDefault k nalezení transakce s daným hashem a typem blockchainu (ChainKind), a výsledek uloží do proměnné blockchainTransaction. Pokud žádná transakce s daným hashem a typem blockchainu není nalezena, funkce vrátí null.

Následně funkce kontroluje, zda nalezená transakce není null a zda sloupec DeletedAt obsahuje hodnotu null (tj. transakce nebyla smazána). Pokud tomu tak je, funkce vrátí nalezenou transakci.

MarkUnattempted

Tato funkce MarkUnattempted slouží k označení transakce v blockchainu (BlockchainTransaction).

Funkce používá přímé SQL dotazy a výrazy @UpdatedAt, @AttemptedAt a @Id jsou použity jako parametry v dotazu.

Funkce nejprve vytvoří dotaz SQL s parametry. Tento dotaz aktualizuje hodnotu sloupce UpdatedAt na aktuální datum a čas a nastaví hodnotu sloupce AttemptedAt na hodnotu null v tabulce BlockchainTransactions pro řádek s konkrétním Id, pokud sloupec DeletedAt obsahuje hodnotu null (tj. řádek nebyl již smazán).

Následně funkce získá aktuální datum a čas pomocí statické metody UtcNow třídy DateTimeOffset a nastaví hodnotu sloupce AttemptedAt na null.

Nakonec funkce provede dotaz SQL na databázi pomocí metody ExecuteSqlRawAsync.

GetUnattempted

Tato funkce GetUnattempted slouží k získání seznamu transakcí v blockchainu (BlockchainTransaction) z databáze, které nebyl proveden pokus o vykonání.

Funkce používá metodu Where pro filtrování transakcí, které mají hodnotu sloupce AttemptedAt rovnu null, hodnotu sloupce DeletedAt rovnu null (tj. nebyly smazány) a mají požadovaný typ blockchainu (ChainKind). Následně funkce vrací seznam transakcí, které splňují podmínky výše.

GetFailed

Tato funkce GetFailed slouží k získání seznamu transakcí v blockchainu (BlockchainTransaction) z databáze, které selhaly při odesílání na blockchain.

Funkce používá metodu Where pro filtrování transakcí, které mají hodnotu sloupce Succeeded rovnu false (tj. selhaly), hodnotu sloupce DeletedAt rovnu null (tj. nebyly smazány) a mají požadovaný typ blockchainu (ChainKind). Následně funkce vrací seznam transakcí, které splňují podmínky výše.

GetAll

Tato funkce GetAll slouží k získání seznamu všech transakcí v blockchainu (BlockchainTransaction) z databáze.

Funkce používá metodu Where pro filtrování transakcí, které mají hodnotu sloupce DeletedAt rovnu null (tj. nebyly smazány). Následně funkce vrací seznam všech transakcí, které splňují podmínky výše.

GetById

Tato funkce GetById slouží k získání transakce v blockchainu (BlockchainTransaction) z databáze pomocí konkrétního Id.

Funkce používá metodu SingleOrDefault pro získání jediné transakce, která má hodnotu sloupce Id rovnu zadanému Id a hodnotu sloupce DeletedAt rovnu null (tj. nebyla smazána).

Pokud transakce s daným Id neexistuje nebo byla smazána, funkce vrátí hodnotu null. Pokud transakce existuje a nebyla smazána, funkce vrátí tuto transakci.

7.4 Budoucí vývoj databáze a backendu

Databáze bude nadále rozvíjena v souladu s požadavky, které se v průběhu dalšího vývoje aplikace objeví. V současné době byl již vytvořen datový model, migrace a přístup k datům pomocí Repository vzoru. V dalším kroku bude třeba vytvořit služby (Services), API rozhraní a přehledné testovací prostředí pro tyto endpointy. Dále bude nutné zajistit zabezpečený přístup k databázi využitím moderních cloudových technologií. Aktuální stav databáze slouží pouze jako testovací prostředí pro vývoj aplikace a bude postupně rozšiřován a vylepšován v souladu s požadavky a potřebami aplikace.

8 TVORBA WEBOVÉ MINTING APLIKACE

Tato kapitola se zaobírá vytvořením webové minting aplikace. Tato aplikace bude umožnovat uživatelům získat vlastní token, který poté bude sloužit jako vstupní herní předmět. Nejprve proběhla analýza požadavků, ve které došlo k definici jednotlivých funkčních a nefunkčních požadavků. Následně proběhl design pro desktop, a to v aplikaci Figma. Tento design je popsaný v této kapitole. Následovalo generování grafických komponentů pro celý projekt, a to za využití umělé inteligence Midjourney. Poté proběhl základní popis implementace důležitých částí.

8.1 Analýza požadavků

Tato kapitola se zabývá popisem všech funkcionalit, které by měla webová minting aplikace obsahovat. Tyto funkcionality byly předem diskutovány na základě potřeb budoucí herní aplikace. Všechny tyto funkcionality jsou navržené tak, aby poskytnuli uživateli co nejlepší zážitek z webového mintování.

Přehledné uživatelské rozhraní

Jedná se o požadavek, který je klíčový pro uživatelský zážitek z aplikace. Tento požadavek by měl poskytnout přehledné a jasné uživatelské rozhraní, které umožní jednoduchý přístup k ostatním funkcionalitám, a poskytne informaci o tom, jak aplikace funguje. Za nejdůležitější části tohoto požadavku je nutno vzít v potaz jednoduché ovládání, jasný design a dobře organizované informace.

Zobrazení tří základních NFT pomocí grafických prvků

Tento požadavek umožnuje uživatelům zobrazit tři základní NFT pomocí grafických prvků v uživatelském rozhraní. Grafické prvky by měly být kvalitní a jasně prezentovat daná NFT, aby uživatelé mohli snadno identifikovat o jaká NFT se jedná. Grafické prvky by měli být taky vizuálně atraktivní, aby přilákali nové uživatele. Pro použití těchto grafických prvků by měli být využité obrázky ve správném formátu tak, aby poskytovali rychlé načítání a dobrou kvalitu.

Možnost mintovat jedno ze tří uživatelských NFT

Tato funkcionalita, má umožnit uživateli vytvořit si vlastní NFT z nabízených tří základních. Uživatel si bude moci vybrat konkrétní NFT, a po jeho výběrů na něj vyskočí transakce. Po

dokončení této transakce by tento token měl obdržet na svoji peněženku. Tato funkcionalita by měla obsahovat jak přihlašování Metamasky, tak připojení k blockchainu.

Přesměrování na informační stránku

Tento požadavek umožnuje uživatelům přesměrování na informační stránku pro získání dalších informací o samotné aplikaci a jednotlivých NFTs. Přesměrování by mělo být snadno dostupné a intuitivní pro uživatele. Například jako tlačítko anebo vhodně umístěný odkaz v uživatelském rozhraní.

8.1.1 Funkční požadavky

V aplikaci existují čtyři různé funkčních požadavky, které pokrývají všechny funkcionality aplikace. Tyto požadavky lze sledovat na Obrázek 15.

Obrázek 15 Funkční požadavky minting aplikace

REQ1 přehledné uživatelské rozhraní

Jedná se o funkční požadavek, který slouží k vytvoření uživatelského zážitku, který musí být součástí celé aplikace. Uživatel by měl mít hezké a přehledné prostředí, které mu budou poskytovat grafické elementy. Toto prostředí by mělo splňovat jednoduchost a interaktivitu, aby uživatel neztratil zájem o aplikaci.

REQ2 zobrazení tří základních NFT pomocí grafických prvků

Funkční požadavek zobrazení tří základních NFT pomocí grafických prvků, slouží k vytvoření tří piedestalů, na kterých budou umístěné tři základní NFT, a to jako grafické elementy, ze kterých si uživatel bude moci vybrat. Tyto NFT by měli přehledně zobrazovat to, co uživatel získá a co může očekávat. Po najetí na NFT bude také zobrazen základní popis.

REQ3 možnost mintovat jedno ze tří uživatelských NFT

Tento funkční požadavek umožní uživateli mintovat jedno ze tří vybraných NFT, a to pomocí kliknutí na tlačítko, které bude umístěno vždy pod jednotlivými NFT na piedestalu. Uživatel klikne na tlačítko a vyskočí mu potřebná transakce pro získání NFT za pomocí Metamasku. Po potvrzení této transakce na blockchainu a zaplacení poplatku a ceny NFT, se mu nový předmět objeví na peněžence.

REQ4 přesměrování na informační stránku

Jedná se o funkční požadavek, který má za úkol poskytnout uživatelům možnost navštívení hlavní stránky, na které získají další potřebné informace, týkající se projektu. Na stránce se bude nacházet odkaz anebo tlačítko, které uživatele přesměruje na informační stránku aplikace.

8.1.2 Nefunkční požadavky

V aplikaci se nachází dva nefunkční požadavky. Všechny tyto nefunkční požadavky podrobně popisují omezení celého systému. Tyto nefunkční požadavky zobrazuje Obrázek 16.

Obrázek 16 Nefunkční požadavky minting aplikace

NREQ1 bezpečnost

U tohoto požadavku je kladen důraz na zabezpečení aplikace při použití Metamasku, což zahrnuje hlavně ochranu proti podvržení adresy tzv proti spoofingu a to takovým způsobem, že uživatel pro získání NFT podepisuje transakci svým privátním klíčem.

NREQ1 Desktop-only aplikace

Jedná se o omezení aplikace, protože není v plánu dělat mobilní design z důvodu, že Metamask nemá integraci do mobilního prohlížeče.

8.2 Design

Tato kapitola je zaměřená na důležitou součást minting webové aplikace, jedná se popis jejího designu. Celý design byl, respektive jeho grafické části byl generován pomocí služby Midjourney, která využívá strojové učení Dal-E. První část popisuje vizuální koncept a myšlenku samotného designu aplikace. V další části je popis generování grafických prvků pomocí Midjourney včetně všech textových příkazů, které jsou třeba pro docílení podobného výsledku. Ve finální části designu je zobrazen React tree, který je v případě této aplikace velmi jednoduchý.

8.2.1 Vizuální koncept

Tato část aplikace je velmi důležitá, neboť design může odradit uživatele od nákupu potenciálního NFT. Z tohoto důvodu byl kladen důraz na vytvoření atraktivního designového návrhu, který bude odrážet herní prostředí.

Při navrhování vizuálního konceptu byl primárně kladen důraz na požadavky aplikace, ale také na splnění základních designových požadavků.

Celý koncept je laděn do animovaného stylu, což odpovídá informační stránce a vyvolává v uživateli pocit hry. Na samotné stránce se nachází jen několik grafických prvků, aby byly splněny funkční požadavky.

Hlavním designovým prvkem stránky je animovaný industriální styl. Tento styl by měl být uplatněn napříč celou aplikací a vytvářet tak dojem "laboratoře".

Vizualní koncept je také součástí aplikace, konkrétně v nástroji Figma, kde byl návrh celé aplikace vytvořen. Je třeba brát v úvahu, že návrh je pouze informativní a má sloužit jako podklad pro budoucí vývoj.

8.2.2 Grafické prvky

Grafické prvky využité v této aplikaci, byly všechny vytvořené pomocí strojového učení MidJourney, který využívá DALL-E. U všech těchto prvků byl kladen důraz na to, aby byl zachován industriální styl, tak jak je vyžadováno v designu.

Laboratoř

Za cíl bylo vytvořit pozadí stránky, které bude odpovídat industriálnímu stylu, konkrétně laboratoře, cílem bylo, aby obsahovalo co nejvíc volného prostoru z důvodu možnosti umístění dalších grafických elementů. Zde jako vstup pro generování sloužil obrázek industriálního prostředí doplněný o textový příkaz „industry room“. Výsledek je možno pozorovat na Obrázek 17.

Obrázek 17 Laboratoř

Monsters

Cílem bylo vytvořit monstra, která budou sloužit jako placeholder pro budoucí NFT, tyto monstra měla vždy jako předlohu první vygenerovaný obrázek, který lze vidět na Obrázek 18, toho bylo docíleno pomocí textového příkazu „Adorable cartoon owl, white, pastel blue and pink, dramatic lighting, illuminated face 85mm“

Obrázek 18 Předloha pro monstra

Další monstrum, které bylo vygenerované, byla žirafa, jako vstup sloužila předloha a textový příkaz „Turquoise Giraffe Hatdramatic lighting, illuminated face 85mm, adorable, cartoon“, výsledek je možné sledovat na Obrázek 19.

Obrázek 19 Monstrum žirafa

Třetí monstrum, co bylo vygenerované byl hroch, pro jeho vytvoření bylo potřeba použít originální předlohu s textovým příkazem „Salmon Hippopotamus Scarfdramatic lighting, illuminated face 85mm, adorable, cartoon“ výsledek generování zobrazuje Obrázek 20.

Obrázek 20 Monstrum hroch

Jako poslední monstrum byl vygenerován varan komodorský, kde zase tak jako u předchozích monster byl jako vstup použit originální obrázek xx. Byl doplněn o textový příkaz „Violet Komodo Dragon Backpackdramatic lighting, illuminated face 85mm, adorable, cartoon“.Obrázek 21 ukazuje výsledek generování.

Obrázek 21 Monstrum varan

Nádoby

Cílem bylo vytvořit nádoby, které se hodí do prostředí industriální laboratoře a budou znázorňovat boxy pro jednotlivé monstra, dle jejich rozdělení. První nádoba byla generována

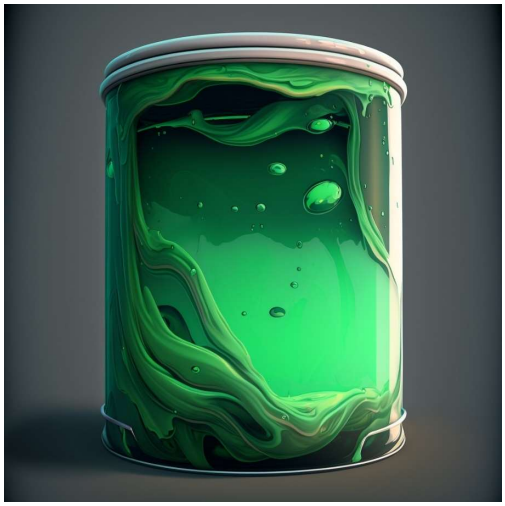
pomocí textového příkazu „Cylidrical glass container that is filled with red fluid“. Výsledek ukazuje Obrázek 22. Tato nádoba poté slouží i jako předloha pro další generování.

Obrázek 22 Červená nádoba

Další nádoba, co byla generována, byla žlutá, pro její vytvoření bylo využito textového příkazu ve znění: „Cylidrical glass container that is filled with yellow fluid“, který byl doplněn u původní obrázek nádoby. Výsledek lze pozorovat na Obrázek 23.

Obrázek 23 Žlutá nádoba

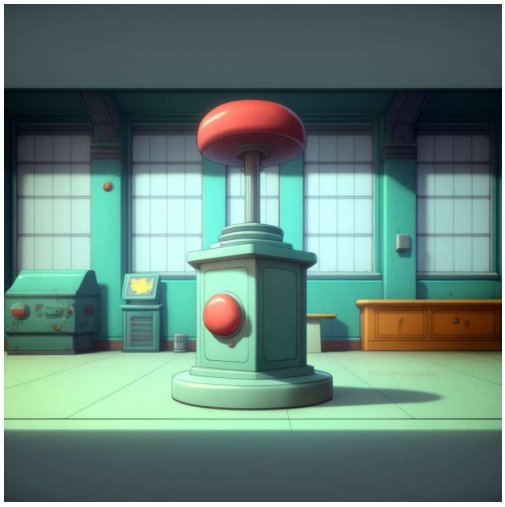
Jako finální, byla generována nádoba v zelené barvě, pro její vytvoření byl využit textový příkaz „Cylidrical glass container that is filled with green fluid“ s přidáním původního obrázku nádoby. Obrázek 24 zobrazuje zelenou nádobu.



Obrázek 24 Zelená nádoba

Pedestal

Cílem bylo vygenerovat grafický prvek, který bude zapadat do prostředí a bude na něj možné umístit monstra a nádoby. Zároveň by měl v uživateli vyvolat dojem toho, že je možné na tento element kliknout. Toho bylo docíleno pomocí textového příkazu „Laboratory pedestal with big red button“, kde jako předloha byl vložen Obrázek 17 Laboratoř .Výsledný pedestal zobrazuje Obrázek 25.



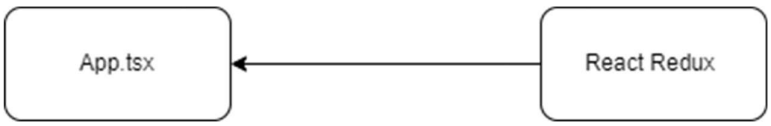
Obrázek 25 Pedestal

8.2.3 Struktura stránky (React tree)

Jedná se o jednostránkovou aplikaci, která má pouze jednu obrazovku a veškeré informace jdou vidět na jednom viewportu. Nejdůležitější částí v aplikaci je React redux, který zde obsluhuje veškerou logiku, co se týká mintování.

V aplikaci se nachází App.tsx, který komunikuje s React redux a to je v podstatě veškerá struktura aplikace.

Díky tomuto uspořádání jsme schopni aplikaci efektivně měnit a nevěnovat jí tolik času vzhledem k tomu, že je pouze dočasná.



Obrázek 26 Struktura minting stránky

Obrázek 26 zobrazuje kompletní strukturu stránky

8.3 Implementace

Tato kapitola se zaměřuje na implementaci a zobrazení důležitých funkcí, které ovládají logiku celé aplikace a umožňují uživatelům projít celým minting procesem. Pro implementaci aplikace byl zvolen jazyk Typescript kvůli jeho silné typové kontrole, která zajišťuje vyšší bezpečnost aplikace. Dále byl využit ESLint, který kontroluje správnost zápisu kódu, což spolu s jazykem Typescriptu vede k psaní bezpečného kódu. V projektu byly použity různé knihovny, které jsou kompatibilní s verzí React 18.2.0 a vyšší. Mezi tyto knihovny patří například react-redux, který slouží k řízení globálních stavů. Dále byl použit gsap verze 3.11.4 a vyšší, který nám umožňuje vytvářet pokročilé animace, a pro stylování byl použit styled components verze 5.3.6 a vyšší, což nám umožňuje znovu používat již definované styly. Jako poslední důležitou knihovnu bychom měli zmínit web3 verze 1.8.2 a vyšší, který nám umožňuje přistupovat k blockchainu a provádět transakce.

8.3.1 Blockchain react-redux

Tato komponenta, stejně jako ostatní React Redux komponenty, je rozdělena na action creators, action types, actions a blockchain reducer. Zde se nachází zásadní část tohoto blockchain reduxu – výchozí soubor stavů aplikace. Tento soubor se skládá ze šesti stavů:

Loading – nastaven na false, určuje, zda se načítá uživatelovo připojení.

Account – nastaven na null, slouží k uložení uživatelské adresy v případě přihlášení.

SmartContract – nastaven na null, slouží k uložení adresy kontraktu, ke kterému je uživatel připojen.

GasPrice – nastaven na 0, slouží k získání aktuálního gasu pro výpočet přesné hodnoty transakce.

Web3 – proměnná, do které se má ukládat aktuální stav Web3, aby bylo možné interagovat s blockchainem.

ErrorMsg – ve výchozím stavu prázdný; do něj se nastavují případné chyby, které můžeme zobrazovat uživatelům, například špatně zvolený blockchain.

V aplikaci se nachází čtyři stavy, které mohou nastav při volání tohoto reduceru, jedná se o stavy

CONNECTION_REQUEST

Tento stav slouží k tomu, aby si vyžádal připojení k blockchainu a na základě něj se změnily potřebné hodnoty, aby aplikace poznala, že se uživatel pokouší o připojení.

CONNECTION_SUCCESS

Tento stav slouží k tomu, aby informoval aplikaci o úspěšném připojení uživatele pomocí Metamasky a předala jí veškeré informace o připojeném uživateli.

CONNECTION_FAILED

Tento stav slouží k tomu, aby informoval aplikaci o neúspěšném připojení uživatele pomocí Metamasky a předala jí veškeré informace o tom proč se toto připojení nepovedlo.

UPDATE_ACCOUNT

Tento stav slouží k tomu, aby informoval aplikaci o změně uživatele, tj. například v případě kdy si uživatel vybere jiný účet.

Dále se v této komponentě nachází nejdůležitější část celé aplikace, a to jsou action creators, které obsluhují veškerou logiku a předávají všechny parametry, dle zavolané akce jedná se o tyto funkce:

connectRequest

Tato funkce connectRequest vrací objekt, type. Tento objekt je definován jako ActionType.CONNECTION_REQUEST, což je pravděpodobně nějaká konstanta, která slouží k identifikaci typu akce.

connectSuccess

Funkce connectSuccess slouží k vytvoření akce, která signalizuje uspěšné připojení uživatele k metamasku. Funkce přijmá jeden argument payload, ten zde slouží k předání informací o připojeném uživately. Definuje typ akce CONNECT_SUCCESS.

connectFailed

Funkce connectFailed slouží k vytvoření akce, která signalizuje neuspěšné připojení uživatele k metamasku. Funkce přijmá jeden argument payload typu string, ten zde slouží k předání zprávy, proč se uživateli nepovedlo připojit. Definuje typ akce CONNECT_FAILED.

updateAccountRequest

Funkce updateAccountRequest slouží k vytvoření akce, která signalizuje změnu uživatelovi peněženky v prohlížečovém rozšíření Metamask. Funkce přijmá arugemnt payload, ten zde slouží k předání stejných informací, jako je tomu u connectSuccess. Definuje typ akce UPDATE_ACCOUNT.

updateAccount

Tato funkce updateAccount je tzv. "action creator", což je funkce, která vytváří akci pro Redux store. Funkce bere jeden argument account, což je objekt reprezentující účet, který bude aktualizován.

Funkce však nevrací přímo objekt akce. Místo toho používá tzv. "async action creator" pattern, kdy vrací funkci, která přijímá dispatch funkci jako argument. Tato funkce pak vnitřně volá jiný action creator updateAccountRequest a výslednou akci dispečuje do Redux store pomocí dispatch funkce.

Connect

Tato funkce connect slouží k připojení aplikace k Ethereum blockchainu pomocí webového rozhraní Web3 a především ke kontraktu, který je použit pro komunikaci s blockchainem.

Funkce vrací funkci (asynchronní action creator), která přijímá dispatch funkci jako argument. Tato funkce vnitřně volá action creator connectRequest, který vytvoří akci typu CONNECTION_REQUEST, která signalizuje začátek procesu připojení.

Poté funkce provádí několik asynchronních operací, včetně stažení ABI pro kontrakt (což je popis rozhraní kontraktu v JSON formátu), zjištění aktuální sítě, ověření, zda je nainstalován Ethereum prohlížečový doplněk (Metamask) a získání uživatelského účtu pro daný kontrakt.

Pokud jsou všechny kroky úspěšné, vytvoří se nová instance kontraktu pomocí stažené ABI a adresy kontraktu, a tuto instanci předáváme jako součást objektu akce typu CONNECTION_SUCCESS. Tento objekt akce také obsahuje informace o uživatelském účtu, Web3 instanci a aktuální ceny gas.

Pokud dojde k nějaké chybě, funkce vytvoří objekt akce typu CONNECTION_FAILED a tuto akci dispečuje do Redux store pomocí dispatch funkce. Výsledkem tohoto procesu je tedy akce, která buď signalizuje úspěšné připojení k blockchainu nebo nějakou chybu, která v průběhu tohoto procesu nastala.

8.3.2 Lokální stavy

Aplikace obsahuje několik lokálních stavů, které nám umožnují poskytnout dynamické renderování stránky a uchování všech potřebných stavů pro plynulý chod aplikace. Všechno tyto stavy lze vidět na Obrázek 27.

```
const [animation, setAnimation] = React.useState(animations);
const [monster, setMonster] = React.useState(monsters);
const [canClaim, setCanClaim] = React.useState(false);
const [signatures, SET_SIGNATURES] = React.useState<Signatures>({});
const [whitelisted, SET_WHITELISTED] = React.useState(false);
const [signature, SET_SIGNATURE] = React.useState();
const [time, setTime] = React.useState(0);
```

Obrázek 27 Všechny stavy aplikace

Animation

Tento stav definuje React state animation, který se používá k uchování informace o animaci, která se má přehrát při mintnutí tokenu. State je inicializován pomocí hooku useState, který bere jako argument výchozí hodnotu animations, která je zde definovaná jako objekt tří proměnných nastavené na false.

Monster

Tento stav definuje React state monster, který se zde používá k uchování informace o tom, jaké monstrum se má uživateli po mintu zobrazit. State je inicializován pomocí react hooku useState a jako výchozí argument beru hodnotu monster, která zde uchovává jako objekt složený z bool hodnot.

canClaim

Tento stav definuje React state canClaim, který zde slouží k uchování toho, jestli uživatel ještě může mintovat další tokeny. State je inicializován ve výchozím stavu na hodnotu false a po vykonání aplikační logiky se nastaví potřebný stav.

Signatures

Tento stav definuje React state signatures, který zde slouží k uchování listu signatures, které slouží k určení toho, jestli je uživatel zapsán na whitelist. Výchozí stav je prázdný objekt, do kterého se v průběhu aplikace načtou potřebná data.

Whitelisted

Reactí hook whitelisted, slouží k uchování a nastavení toho, jestli uživatel je whitelisted k tomu, aby mohl mintovat v předprodeji tokenů. O tuto logiku se stará dále aplikace a volá právě SetWhitelisted, aby ověřila jestli adresa má právo mintovat tokeny.

Signature

Jedná se o React hook, který má za cíl uchovat stav konkrétní podpisu ke konkrétní adrese, která má na minting nárok. Slouží zde poté k tomu, aby prokázal listproof do funkce předčasného mintování na whitelistu.

Time

Tento React hook nastavuje aktuální čas proto, aby aplikace mohla rozeznat, kdy minting ještě není vůbec spuštění a někdo kdy běží předprodej tokenů. Tento stav se nastavuje v useeffect.

8.3.3 UseEffects

V aplikaci se nachází dva React hooky useEffect, které slouží k aktualizaci hodnot a provedení kódu při změně určitých proměnných. Tyto hooky fungují jako listenery na určité události.

První z těchto hooků se spouští pokaždé, když se změní hodnota proměnné blockchain.account, která zastupuje uživatelskou adresu. Jeho účelem je zavolat funkci getData(), která získává data z blockchainu a aktualizuje stav aplikace na základě těchto dat.

Druhý hook závisí na proměnné animations a slouží k vykreslení animace a příslušného monstra na základě změněného objektu animations. Existují zde tři různé možnosti, jakým směrem se tento hook může vydat, a to podle toho, jaké monstrum a animace se mají vykreslit.

8.3.4 Využité funkce

V této části jsou popsány všechny zbylé funkce, které mají vliv na aplikaci. Jedná se o důležité funkce, bez kterých by aplikace nemohla fungovat.

getData

Funkce nejprve kontroluje, zda jsou tyto proměnné nastaveny – pokud je uživatelská adresa prázdná nebo smart kontrakt není definován, funkce se neprovede a nedojde ke zpracování dat z blockchainu.

Pokud jsou oba stavy definovány a podmínky projdou, tak se volá metoda MintedByAddress na smart kontraktu. Tato metoda získává informace o tom, zda uživatel již mintoval token pro tuto adresu v minulosti.

Volání metody se provádí pomocí call, což je asynchronní operace. Pokud dojde k chybě, funkce tuto chybu zaznamená v konzoli. Pokud vše proběhne úspěšně, tak se získaná hodnota zkontroluje a pokud je větší než 0, tak funkce nastaví hodnotu canClaim na true, což indikuje, že uživatel má právo nárokovat si token. Pokud je hodnota 0, tak se hodnota canClaim nastaví na false, což znamená, že uživatel již nemá nárok na další token pro tuto adresu.

getGasPrice

Tato funkce getGasPrice slouží k získání aktuální ceny gas na Ethereum blockchainu.

Funkce nejprve získá objekt ethereum z globálního objektu window. Pokud se tento objekt nenajde, funkce vyhodí chybu "Ethereum object not found in window".

Následně se vytvoří nový objekt Web3, který se používá pro interakci s Ethereum sítí.

Poté se volá metoda getGasPrice na objektu web3.eth, která asynchronně získá aktuální cenu gas.Tuto cenu gas funkce vrátí jako výstup.

getTimestamp

Tato funkce getTimestamp slouží k získání časového razítka v sekundách.

Funkce používá vestavěný objekt Date pro získání aktuálního času a metodu getTime, která vrací počet milisekund. Tento počet milisekund se následně dělí 1000, čímž se získá počet sekund. Tento vypočtený čas se následně uloží do proměnné timestamp a uloží se do stavu komponenty pomocí funkce setTime.

getSignatures

Tato funkce getSignatures slouží k získání dat ze souboru signatures.json a nastavení těchto dat do stavu komponenty.

Funkce využívá asynchronní operaci fetch pro získání dat ze souboru signatures.json. Tato operace vrací Promise, který obsahuje HTTP odpověď na dotaz. V této funkci se čeká na dokončení této operace pomocí klíčového slova await.

Poté se získaná odpověď převede na JSON formát pomocí metody json(). Tento krok také probíhá asynchronně a opět se používá klíčové slovo await.

Nakonec se získané JSON data nastaví do stavu komponenty pomocí funkce SET_SIGNATURES.

getWhitelisted

Tato funkce getWhitelisted slouží k zjištění, zda je uživatelská adresa blockchain.account na whitelistu a nastavení příslušného stavu komponenty.

Funkce nejprve nastaví výchozí hodnoty proměnných whitelisted na false a signature na undefined.

Poté se pomocí metody Object.keys() projde seznam klíčů objektu signatures. Pro každý klíč se zkontroluje, zda se shoduje s uživatelskou adresou blockchain.account, a to bez ohledu na velikost písmen. Pokud je shoda nalezena, funkce nastaví proměnnou whitelisted na true a proměnnou signature na hodnotu, která odpovídá tomuto klíči v objektu signatures.

Nakonec se nové hodnoty proměnných whitelisted a signature nastaví do stavu komponenty pomocí funkcí SET_WHITELISTED a SET_SIGNATURE.

privateSaleMint

Tato funkce privateSaleMint slouží k vytvoření transakce pro minting tokenů v předprodeji.

Funkce má tři parametry: signature, což je podpis transakce, gasPrice, což je cena gas v síti Ethereum, a monsterType, což je typ tokenu, který uživatel mintuje.

Nejprve se vytvoří proměnné cost a gasLimit, které obsahují cenu v wei a limit gas pro transakci. Tyto hodnoty jsou načteny ze souboru configTest. Následně se zavolá funkce getData, která získá informaci o tom, zda uživatel má právo mintovat tokeny v předprodeji.

Poté se volá metoda presaleMint na objektu blockchain.smartContract.methods s parametry signature a monsterType. Tato metoda vytváří transakci pro minting tokenů přímo na blockchainu.

Nakonec se volá metoda send na této transakci, která odesílá transakci do sítě Ethereum. Tato metoda má několik parametrů, jako je cena gas, adresa kontraktu (to), adresa uživatele

(from), cena nakupovaných tokenů (value) a limit gas. Při uspěšném dokončení se zavolá funkce setAnimation, proto, aby uživateli vykreslila daný token.

publicSaleMint

Tato funkce publicSaleMint slouží k vytvoření transakce pro minting veřejných tokenů.

Funkce má dva parametry: gasPrice, což je cena gas v síti Ethereum, a monsterType, což je typ tokenu, který uživatel nakupuje.

Nejprve se vytvoří proměnné mintAmount, cost a gasLimit, které obsahují počet nakupovaných tokenů, cenu v wei a limit plynu pro transakci. Tyto hodnoty jsou načteny ze souboru configTest.

Následně se zavolá funkce getData, která získá informaci o tom, zda uživatel již nevymintoval všechny tokeny, na které má právo.

Poté se volá metoda publicSaleMint na objektu blockchain.smartContract.methods s parametry mintAmount a monsterType. Tato metoda vytváří transakci pro minting tokenů.

Nakonec se volá metoda send na této transakci, která odesílá transakci do sítě Ethereum. Tato metoda má několik parametrů, jako je cena gas (gasPrice), adresa kontraktu (to), adresa uživatele (from), cena nakupovaných tokenů (value) a limit gas (gasLimit). V případě uspěšné transakce fukce volá setAnimation pro nastavení příslušné animace.

handleConnectWallet

Tato funkce handleConnectWallet se volá při kliknutí na tlačítko "Connect Wallet" a slouží k připojení uživatele k jeho MetaMasku.

Nejprve se volá funkce dispatch s parametrem connect, což je action creator, který vytváří akci pro Redux store, aby aplikace mohla komunikovat se smart kontraktem.

Poté se volají funkce getTimestamp, getSignatures a getWhitelisted, které slouží k získání časového razítka, podpisů transakcí a informace o tom, zda uživatel může mintovat tokeny. Tyto funkce jsou popsány v předchozích odpovědích.

8.3.5 Dynamické renderování stránky (podmínky zobrazení)

Tato část se zabývá dynamickým renderování stránky na základě podmínek, jedná se o situace, kdy je nutno skrýt a nebo zobrazit nějakou část obsahu v závislosti na podmínkách uživatele. V naší aplikaci se nachází více takových situací, nejdůležitější jsou popsané níže.

Kontrola přihlášení

Podmínka "blockchain.account === "" || blockchain.smartContract === null" ověřuje, zda jsou dvě proměnné "blockchain.account" a "blockchain.smartContract" prázdné nebo null. Pokud alespoň jedna z těchto proměnných není definovaná nebo je prázdná, podmínka se vyhodnotí jako true, což znamená, že určitý obsah na stránce bude zobrazen.

Kontrola, jestli už běží období mintingu

Tato podmínka kontroluje, zda je aktuální čas větší než hodnota 1671454800 (čas mintingu). Pokud je toto tvrzení pravdivé, pak se vyhodnotí první výraz ve výrazu ternárního operátoru, tedy se zobrazí určitý obsah na stránce.

Kontrola, jestli už běží období předprodeje (Whitelist)

Tato podmínka kontroluje, zda je aktuální čas menší než hodnota 1671465600 (čas whitelist mintingu) a zároveň, zda uživatel není ve whitelistu. Pokud oba výrazy platí, podmínka se vyhodnotí jako true a zobrazí se určitý obsah na stránce. Pokud alespoň jeden z výrazů není splněn, podmínka se vyhodnotí jako false a obsah se nezobrazí.

Kontrola, jestli uživatel může ještě mintovat

Tato podmínka kontroluje, zda proměnná "canClaim" má hodnotu true. Pokud ano, zobrazí se určitý obsah na stránce. Má za úkol kontrolovat, jestli uživatel má ještě právo na mintování tokenů.

Generování prvků animace a monster

Tato podmínka se skládá ze dvou ternárních operátorů a vyskytuje se na stránce třikrát v závislosti na dané situaci. Tato podmínka testuje následující proměnné:

animation.animationReedem_x

monster.monster_x

kde x může nabývat hodnoty 1 až 3. Pokud je první proměnná true, na stránce se zobrazí animace. Pokud je první proměnná false, provede se test na druhou proměnnou a pokud druhá proměnná je true, na stránce se zobrazí konkrétní obrázek monstra, které uživatel získal.

9 TVORBA INFORMAČNÍHO WEBU

V této kapitole je uveden celý postup vývoje informačního webu, který bude sloužit pro informování uživatelů o novinkách a plánu herní aplikace. Nejprve proběhla analýza požadavků, došlo k definování funkcí a vytvoření jednoduchého diagramu. Následně byl zvolen jazyk Typescript a framework React a to z důvodu velkého rozšířenosti ve Web3 světe a možnosti znuvupoužitelnosti komponent. Poté proběhl základní design pro desktop, a to v aplikaci Figma. Následovalo generování grafických podkladů, a to pomocí umělé inteligence Midjourney, která využívá DAL-E. Poté proběhla implementace s následným hostingem a zabezpečením.

9.1 Analýza požadavků

V této kapitole jsou popsány všechny funkcionality, které by měla informační webová stránka poskytovat. Tyto funkcionality, byly předem přezkoumány na základě průzkumu ostatních webových stránek, které se zaobírají Web3 herními projekty. Všechny funkcionality jsou navržené tak, aby byly pro uživatele co nejpohodlnější a nebyly mu poskytovány zbytečné informace.

Informace o vývoji aplikace

Jedná se požadavek, který má za úkol poskytnout uživateli relevantní a aktuální informace, týkající se dalšího vývoje aplikace a aktuálních novinek, jako jsou poznámky k opravám atd. Tento požadavek by měl být zahrnut v podobě videa, na který bude vést odkaz z jiné platformy jako je Youtube anebo Twitter. Uživatel by měl být schopen toto video ovládat základním uživatelským rozhraní, dle platformy, ze které toto video bude převzato.

Informace o guildách

Tento požadavek má za cíl, uživateli předat interaktivně informace o dostupných guildách, které se ve hře budou nacházet a podat základní přehled o jejich fungování. Důležité je u tohoto prvku důraz na interakci uživatele a jednoduchého a rychlého poskytnutí informací.

Informace o herních mechanikách

Jedná se o velmi důležitý požadavek, který má za úkol předat hráči dostatečný počet přehledných informací o herních mechanikách, a to konkrétně o herní mechanice Play to earn, tj mechanice, která má hráči přinášet možnost získat finanční prostředky hraním hry. Je důležité, aby tento prvek byl jasný a dostatečně přehledný pro všechny uživatele a zároveň

nezahlcoval uživatele přílišným množstvím informací, které pro ně v začátku nejsou důležité.

Poskytnutí herních záběrů

Tato komponenta by měla být jednoduchá, a měla by mít za cíl poskytnout uživateli relevantní a aktuální informace o hře samotné. Mělo by se tedy jednat o požadavek, který bude zakomponován ve formě videa, na které budou vést odkazy z jiných platforem. Uživatel bude schopen toto video ovládat uživatelským rozhraním definované platformou.

Roadmap

Jedná se o jeden z nejdůležitějších požadavků, které mají za cíl poskytnout uživateli informace o budoucím vývoji projektu. Měl by být kladen důraz na interaktivitu a vyvolat v uživateli dojem, toho, že se jedná o cestu projektem, kterým si projekt musí projít. Je také velmi důležité zobrazit u jednotlivých fází a jejich milníků potřebné informace, které poskytnout dostatečné info o dané částí. U každé části by také mělo být dostatečně jasné, že je možná další interakce.

Sociální sítě

Cílem tohoto požadavku, je vytvořit interaktivní elementy, které poskytnout uživateli možnost plynule přejít na sociální sítě týkající se projektu. Jedná se o socialní sítě Facebook, Instagram, LinkedIn a Tiktok.

Možnost odběru novinek

Tento požadavek, má za úkol poskytnout uživateli možnost, aby se mohl přihlásit k odebírání novinek prostřednictvím emailu. Je důležité klást důraz na design prvku tak, aby uživatel byl motivován tyto novinky odebírat.

Přesměrování na minting aplikaci

Tento jednoduchý požadavek, má uživateli poskytnout možnost se přesměrovat na mintovací aplikaci.

Možnost stažení whitepaper

Cíl tohoto požadavku, by měl být umožnit uživateli zobrazit anebo stáhnout tzv “whitepaper” a to prostřednictvím třetí strany zvané gitbook.

Personalizace pomocí vlastněných NFT

Daný požadavek je velmi důležitý, a to právě z toho důvodu, že ukazuje rozdíl mezi Web2 a Web3 webovou aplikací, má za úkol poskytnout přihlášení krypto peněženky, jako je například Metamask, a pomocí uživatelské adresy (veřejné) a jejího napojení na NFT kontrakt. Naskenovat obsah peněženky a uživatelské NFT, které vlastní v dané hře a na základě těchto NFT personalizovat obsah stránky tak, aby se objevovaly jeho NFT a tím mu zpříjemnili zážitek prohlížení.

9.1.1 Funkční požadavky

V aplikaci existuje jedenáct různých funkčních požadavků. Všechny tyto požadavky dohromady popisují základní fungování aplikace. Tyto požadavky lze sledovat na Obrázek 28.

Obrázek 28 Funkční požadavky

REQ1 guilds

Jedná se o funkční požadavek, který slouží k zobrazení záložky guilds, která bude součástí hlavní stránky. Uživatel po kliknutí na tlačítko guilds, bude přesměrován na danou sekci, která se nachází na stránce. Tato sekce bude mít specifický design, který bude obsahovat tři

grafické elementy se kterými uživatel může integrovat pomocí najetí tlačítka myši, kde tato interakce vyvolá informaci o dané guildě.

REQ2 roadmap

Jedná se o funkční požadavek, který zobrazuje sekci s roadmap. Uživatel by měl být schopen po najetí na sekci vidět jednoduchou animaci, která bude mít specifický design odpovídající celé stránce. Na této sekci by se mělo nacházet několik prvků, které budou znázorňovat jednotlivé cíle projektu, kde každý cíl bude mít svůj design. Po najetí na tento cíl, by se měla uživateli vyvolat animace, která poskytne informace k danému cíli. Zároveň by pod každým cílem, měla být informační ikona, která upozorňuje na to, že s elementem se dá interagovat.

REQ3 gameplay

Jedná se o funkční požadavek, který má za úkol zobrazit gameplay, který je součástí hlavní stránky. Po kliknutí je uživatel přesměrován na sekci GamePlay, který bude mít daný grafický design. V tomto grafickém designu bude zakomponováno video, které bude poskytovat uživatelského ovládání, integrované službou youtube.

REQ4 whitepaper

Jedná se o funkční požadavek na zobrazení whitepaper. Uživatel po kliknutí na záložku whitepaper bude přesměrován na stránku gitbook, kde bude poskytnutý whitepaper k projektu.

REQ5 minting

Tento funkční požadavek slouží k tomu, aby přesměroval uživatele na minting aplikaci. Uživatel po kliknutí na záložku minting bude přesměrován na příslušnou doménu, která bude sloužit jako minting aplikace.

REQ6 Tlačítko na připojení Metamasku

Tento funkční požadavek by měl zajišťovat tlačítko pro přihlášení pomocí krypto peněženky Metamask. Uživatel po interakci s tímto tlačítkem by měl být vyzván prohlížečovým rozšířením metamask, pomocí kterého může dokončit přihlášení ke stránce a povolit tak daný přístup k jeho veřejné adrese. Tlačítko bude mít předem daný specifický design.

REQ7 Funkce na načtení dat z peněženky NFT a následné generování obrázků

Jedná se o funkční požadavek pro naskenování dat z dané NFT kolekce. Systém by měl zkontrolovat, jestli uživatelova veřejná adresa vlastní nějaké NFT dané kolekce, a pokud

tokeny budou nalezeny, nahradí se původní designové prvky právě těmito NFT tak, aby byly uživateli poskytnuto personalizované prohlížení stránky.

REQ8 Video o vývoji

Jedná se o funkční požadavek, který má za úkol zobrazit video o vývoji, které je součástí hlavní stránky. Video bude obaleno ve specifickém grafickém designu, které bude poskytovat uživatelského ovládání, integrované službou YouTube.

REQ9 Play and earn mechanismus

Tento funkční požadavek má za cíl zobrazit sekci Play and Earn. V této sekci budou umístěna dvě velká designová okna, která uživatelům poskytnou dostatečné množství informací o tom, jak funguje mechanismus "play to earn", aniž by jim byly poskytnuty příliš podrobné informace.

REQ10 Odkazy na sociální sítě

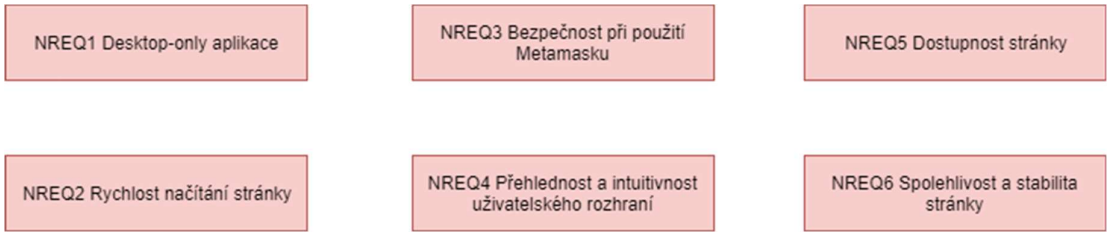
Jedná se o funkční požadavek, který má za úkol poskytnout uživatelům možnost navštívení sociálních sítí týkajících se projektu. Na stránce se bude nacházet sekce, kde budou poskládané ikony, které budou sloužit jako odkaz na čtyři sociální sítě, Instagram, Facebook, TikTok, LinkedIn.

REQ11 Odebírání novinek

Jedná se o funkční požadavek, který má za cíl poskytnout uživateli možnost odebírat novinky pomocí emailu. Na stránce se bude nacházet sekce, která bude poskytovat designové zpracování a vstupní formulář pro možnost vyplnění emailu a přihlášení se k odběru novinek. Tento odběr bude následně zaslán na SMTP klienta, který tento vstup zpracuje a uživateli zašle email o potvrzení přihlášení. Zároveň informuje aplikační email o tom, že zde nový uživatel přihlásil odběr novinek.

9.1.2 Nefunkční požadavky

V aplikaci se nachází šest nefunkčních požadavků. Všechny tyto nefunkční požadavky podrobně popisují omezení celého systému. Tyto nefunkční požadavky zobrazuje Obrázek 29.



Obrázek 29 Nefunkční požadavky

NREQ1 Desktop-only aplikace

Jedná se o omezení aplikace, protože není v plánu dělat mobilní design, z důvodu, že Metamask nemá integraci do mobilního prohlížeče.

NREQ2 Rychlost načítání stránky

Tento nefunkční požadavek zahrnuje doby, za kterou se stránka načte a zobrazí uživateli. Cílem je minimalizovat dobu načítání webové stránky tak, aby byla pohodlnější pro uživatele a její načtení netrvalo déle než 15 sekund.

NREQ3 Bezpečnost při použití Metamasku

U tohoto požadavku je kladen důraz na zabezpečení aplikace při použití Metamasku, což zahrnuje i ochranu proti podvržení adresy.

NREQ4 Přehlednost a intuitivnost uživatelského rozhraní

Tento nefunkční požadavek se zaměřuje na design a uspořádání uživatelského rozhraní. Má za cíl vytvořit jednoduché a intuitivní prostředí, kde uživatel dostane všechny potřebné informace.

NREQ5 Dostupnost stránky

Tento požadavek zajišťuje, že aplikace bude přístupná a dostupná pro uživatele ve většině částí světa. To zahrnuje i zvolení vhodného providera na hosting, tak aby měl dobře rozvržené servery.

NREQ6 Spolehlivost a stabilita stránky

Tento požadavek se zaměřuje na odolnost aplikace vůči chybám a schopnosti udržet chod i v případě většího nárustu uživatelů anebo chvilkové zátěže.

9.2 Design

Tato kapitola se zaměřuje na velmi důležitou část projektu, a to je design samotné aplikace, je nutno podotknout, že celý design, co se týká grafické části je generativní, a to pomocí strojového učení Dal-E. V první části je popsán vizuální koncept a inspirace od ostatních aplikací a její celkový vizuální cíl. Poté následuje popis generování grafických prvků a jejich textový popis, který byl třeba proto, aby generování bylo uspěné. Finální částí designu, je na návrh React tree, tj struktury celého webu a jak mezi sebou budou jednotlivé části komunikovat.

9.2.1 Vizuální koncept

Jedná se o velmi důležitou část aplikace, protože na ní závisí velká část návštěvnosti uživatelů. Proto byl také kladen důraz na prozkoumání největších herních aplikací v teoretické částí a z nich byl udělán výtah designu, které se odráží v návrhu této aplikace.

Při designu vizuálního konceptu, byl kladen důraz hlavně na požadavky aplikace, ale také na to, že se jedná o stránku pro herní aplikaci, tudíž je nutno splnit dynamiku a kreativitu designu tak, aby uživatele udržela na stránce. Z tohoto důvodu, sloužila jako hlavní grafická předloha designu právě zmíněná největší hra Axie Infinity. Jejich stránku lze navštívit na tomto odkazu [67]. Z tohoto designu vznikla inspirace herních textur a obecný návrh umístění některých elementů. Je však důležité podotknout, že design sloužil pouze na částečnou inspiraci, tak jako ostatní velké hry jako je například SandBox.

Vizuální koncept je laděn do animovaného stylu proto, aby v uživatelích nabudil pocit hry. Na stránce se také nachází mnoho grafických elementů a interaktivních objektů, aby stránka nejen předala informace, ale také uživatele zabavila po dobu prohlížení.

Primárním cílem bylo vytvořit prostředí, ve kterém uživatel může projít různá prostředí, která slouží jako pozadí stránky, a to od prostředí mraků, přes hory, až po písečné pozadí, kde tato stránka bude končit.

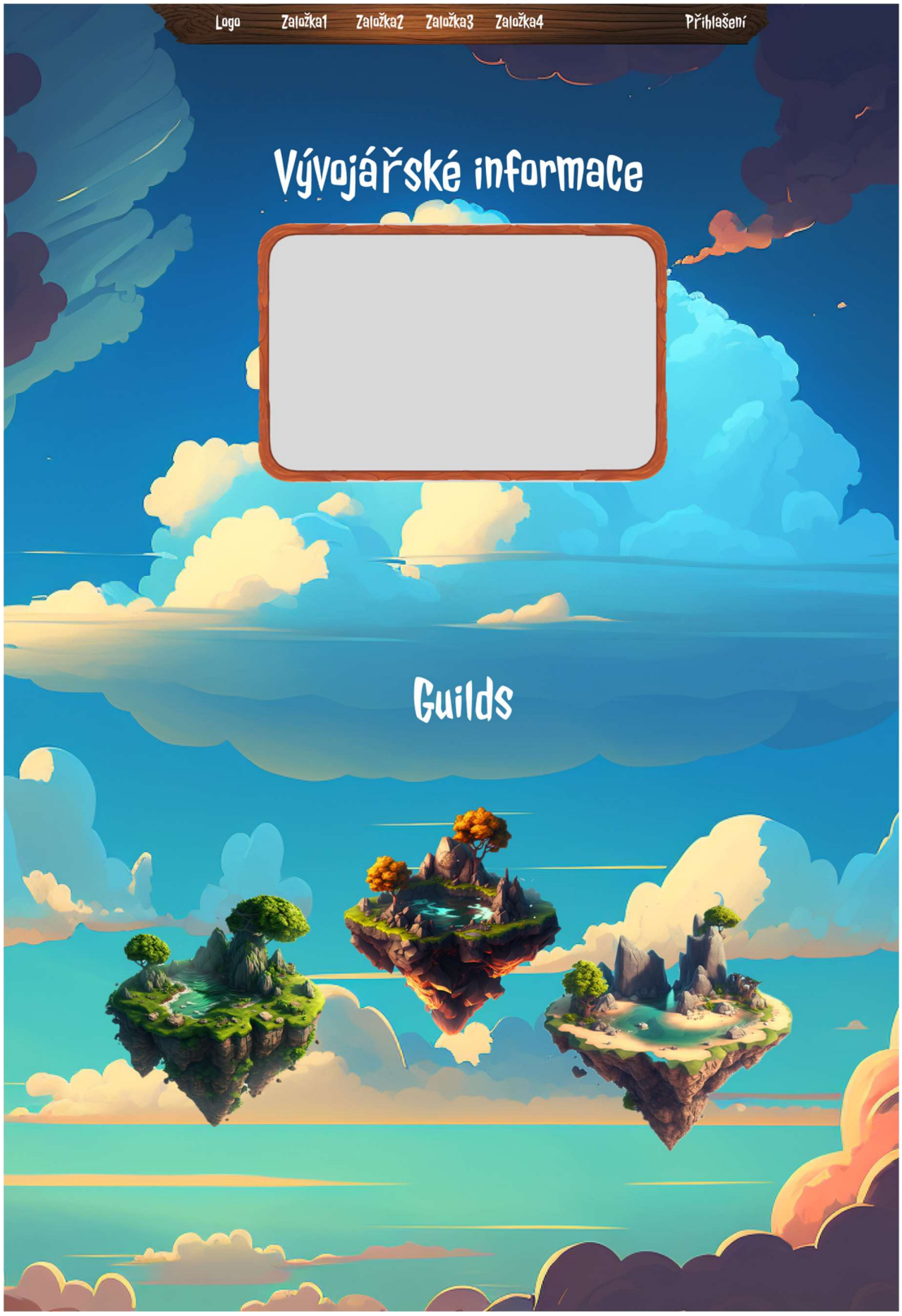
Hlavní texturou, která je zakomponována do stránky, je textura dřeva. Tato textura by měla být využita jak pro hlavní navigační menu, které by mělo představovat dřevěnou desku, tak pro další grafické elementy, jako jsou právě rámečky pro texty anebo rámečky pro integrovaná videa. Důležitou součástí je také zahrnout právě daná pozadí a textury elementů tomu přizpůsobit.

Kromě popisu textur a pozadí, je důležité zmínit i volbu fontu, který byl použit v této aplikaci. Při výběru fontu byl kladen důraz na to, aby byl srozumitelný a čitelný pro uživatele, ale zároveň aby ladil s celkovým vizuálním konceptem a vytvářel dojem herního prostředí. Byl zvolen moderní a dynamický font, který evokuje dojem herního prostředí. Tento font je také vhodný pro zobrazování titulků a důležitých textů v aplikaci, aby byly co nejvíce nápadné a přitahovaly pozornost uživatele.

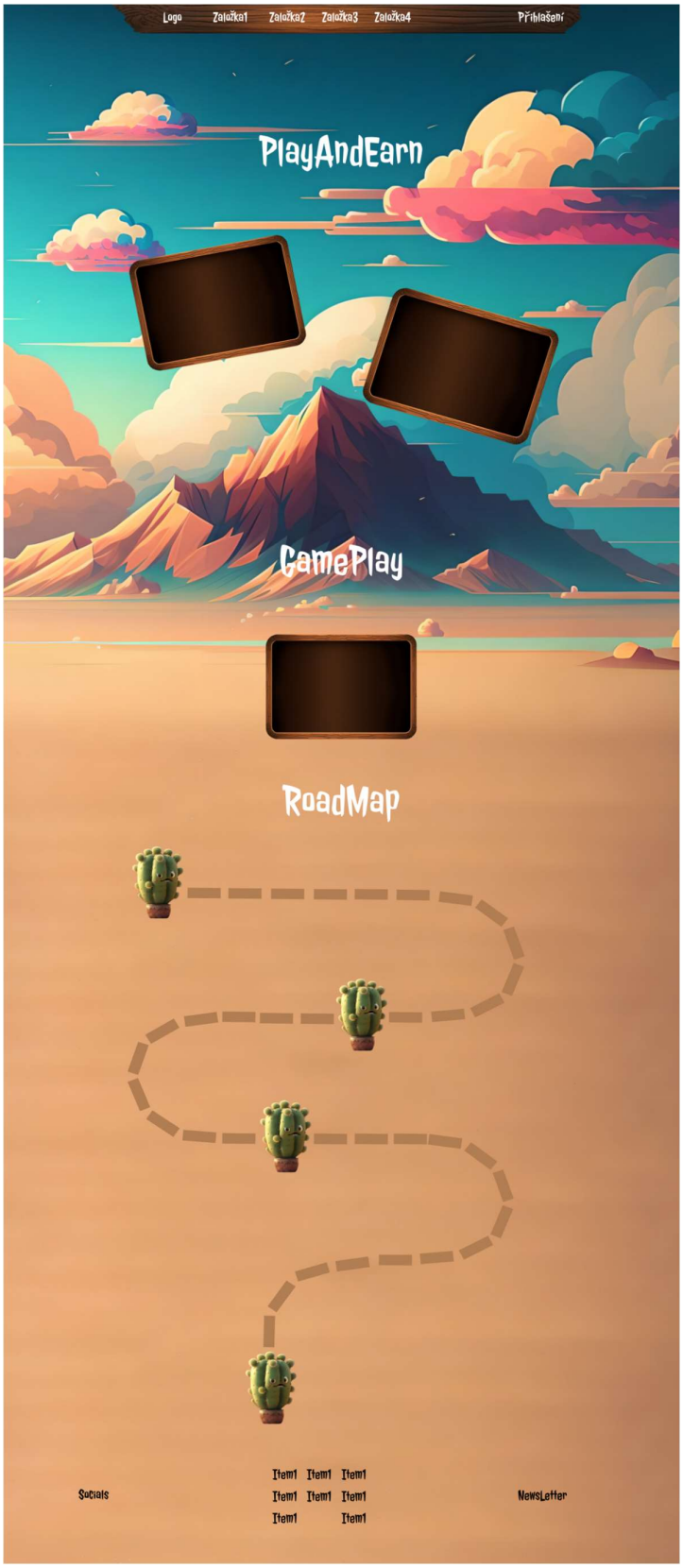
Součástí je také vizuální koncept v aplikaci Figma, kde byl celý tento návrh aplikace vytvořen, je třeba podotknout, že na tento návrh již obsahuje některé textury, které jsou detailněji popsány v další kapitole.

Obrázek 30 ukazuje hrubý návrh hlavní část stránky, která má zobrazovat uživateli základní informaci o hře, jako je právě vývojářské video obalené v rámečku, která odpovídá globální textuře projektu anebo právě Guildy, které jsou zde vyobrazeny jako létající ostrovy, tak aby zapadali do pozadí mraků. Je zde také zobrazen hlavní element navigačního baru, který je vyobrazen jako dřevěná deska a obsahuje všechny elementy, které vycházejí ze vstupních požadavků hlavní stránky, tento element by měl být dostupný i na druhé části stránky.

Obrázek 31popisuje druhou část stránky, která zobrazuje návrh druhé částí informační stránky, na kterou se uživatel dostane v moment, kdy dojde na konec první. Tento obrázek ukazuje popis elementu herní mechaniky play and earn, který je zde vyobrazen jako dvě dřevěné tabule, které opět vychází z designu celé stránky a globálně nastavené textury dřeva. Poté následuje sekce RoadMap, kde se nachází cesta v písku, právě pomocí kroků, tento element by měl být animovaný. Jednotlivé cíle mají být dostatečně odlišené a cesta by měla vést skrze ně. Ve spodní části stránky se nachází patička, a v té jsou obsaženy sociální sítě dle funkčních požadavků a možnost odebírání newsletteru.



Obrázek 30 Informační webová stránka část I



Obrázek 31 Informační webová stránka část II

9.2.2 Grafické prvky

Grafické prvky využité pro stránku byly všechny generované pomocí strojového učení DALL-E, konkrétně pomocí služby MidJourney. U všech těchto prvků byl kladen důraz na to, aby byl dodržen design stránky a její návrh.

Pozadí pro první část stránky

Za cíl bylo vytvořit, pozadí, které bude obsahovat mraky a oblohu v animovaném stylu. Toho bylo docíleno pomocí chatového zadání v následujícím znění „Background on which will be the Sky, with few clouds in animated style“. Obrázek 32 ukazuje finální grafický prvek.

Obrázek 32 Pozadí první část stránky

Pozadí pro druhou část stránky

Za cíl bylo vytvořit pozadí stránky, které bude navazovat na pozadí první částí, a bude obsahovat hory a poušť, tak jak bylo popsáno v designu. Toho bylo docíleno pomocí chatového zadání, kde jako vstup byl Obrázek 32 a z něj měl za úkol vygenerovat navazující pozadí s klíčovou vlastností „Mountains“.Obrázek 33 popisuje finální grafický prvek.

Obrázek 33 Pozadí druhá část stránky

Navigační bar

Tento grafický element, byl generován s cílem vytvořit dřevěnou desku, na kterou by se dali umisťovat navigační body, toho bylo docíleno pomocí chatového zadání ve znění „Light wooden board animated“, které sloužilo k vygenerování dřevěného rámu, jak lze pozorovat na Obrázek 34, z tohoto rámu následně byla využitá jen část, která právě slouží jako navigační bar.

Finální element lze pozorovat na Obrázek 35.

Obrázek 34 Dřevěný rám

Obrázek 35 Navigační bar

Příšery na stránce

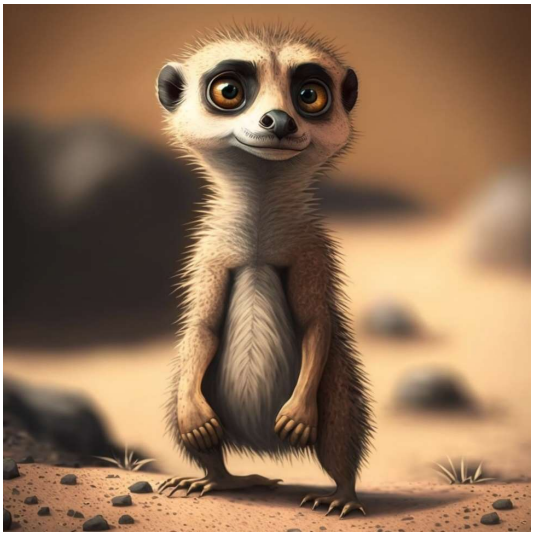
Na stránce jsou generovány také grafické prvky tykajících se příšerek, které jsou rozmístěné po stránce, právě z důvodu toho, aby stránka byla živější. Generován byl drak, a to pomocí textového zadání „Flying dragon animated“, tento finální grafický ukazuje Obrázek 36.

Obrázek 36 Drak I

Poté byl vytvořen další drak, avšak jiného typu, a to pomocí textového zadání „Yellow animated flying dragon“. Finální výsledek ukazuje Obrázek 37.

Obrázek 37 Drak II

Finální příšera na stránce je surikata, která byla vygenerována pomocí textového zadání ve znění: „Animated meerkat cartoon“, tento finální výtvor lze sledovat na Obrázek 38.



Obrázek 38 Surikata

Cíle roadmapy

Cíle roadmapy byly generovány tak, aby byly výrazné a zapadaly do pouště a do designu pozadí. První ze zmíněných je cedule, která byla generována pomocí textového příkazu „Wooden sign“. Obrázek 39 ukazuje výsledný grafický prvek.

Obrázek 39 Cedule

Poté následuje kaktus, který byl generován pomocí textového příkazu „Cartoon cactus “. Finální podobu popisuje Obrázek 40.

Obrázek 40 Kaktus

dalším cílem je totem, který je generován pomocí textového příkazu „Buffalo skull animated “ a je vyobrazen na Obrázek 41.

Obrázek 41 Totem

Finálním milníkem je grafický element pokladu, který byl generován pomocí textového příkazu „Animated treasure“. Popisuje jej Obrázek 42.

Obrázek 42 Poklad

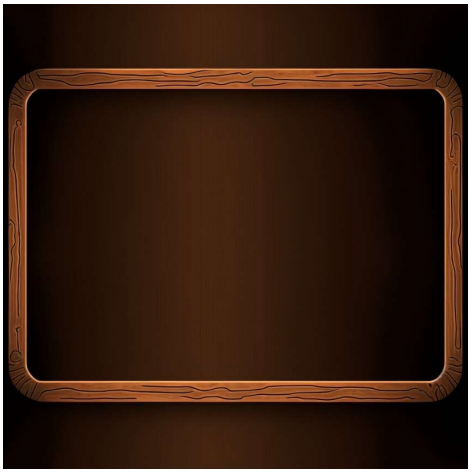
Newsletter

Za cíl bylo vytvořit prvek, který bude uživateli jasně říkat, že se jedná o newsletter, a že zde může odebírat novinky o projektu. Grafický prvek byl vygenerován pomocí textového příkazu „Mailbox animated“ a lze sis jej prohlédnout na Obrázek 43.

Obrázek 43 Mailbox

Video rámeček

Tento grafický element má za cíl zvýraznit video pomocí dřevěného rámu, tak jak bylo navrženo v kapitole design. Tento video rámeček byl vygenerován pomocí textového příkazu „Animated light wood border“. Obrázek 44 ukazuje finální výsledek.



Obrázek 44 Video rámeček

Textová tabule

Cílem tohoto generování bylo vytvořit textovou tabuli, která by odpovídala designu celé stránky, pro toto byl využit grafický prvek Obrázek 44 Video rámeček s pozměněným pozadím.

Létající ostrovy

Tento grafický element, byl vytvořen tak, aby zapadl do pozadí na první části stránky. I z tohoto důvodu byly vygenerovány tři létající ostrovy, které mají odlišovat jednotlivé guildy. Pro každý z těchto ostrovů, byl jako ukázkový obrázek předložen Obrázek 45 Létající ostrov I, proto aby byly podané konzistentní výsledky.

- **Létající ostrov I**

  Tento ostrov, byl generován ve stylu přírody, proto byl zvolený následný textový příkaz „Flying island animated, grass“. Tento ostrov lze pozorovat na Obrázek 45.

Obrázek 45 Létající ostrov I

- **Létající ostrov II**

  Druhý ostrov, byl generován v lávovém a kamenitém stylu, i proto bylo pro generování zvoleno textové zadání „Flying island animated, lava, rock“. Obrázek 46 ukazuje druhý ostrov.

Obrázek 46 Létající ostrov II

- **Létající ostrov III**

  Poslední ostrov byl generován, jako pískový, proto aby bylo dosáhnuto požadovaných výsledků byl zvolen následující textový příkaz: „Flying island animated, sand, rocks“. Obrázek 47 popisuje písečný ostrov.

Obrázek 47 Létající ostrov III

Ikony

Tyto grafické prvky nebyly jediné generovány pomocí MidJourney, ale byla využita veřejně dostupná knihovna ikon. Všechny ikony lze najít na Obrázek 48.

Obrázek 48 Použité ikony

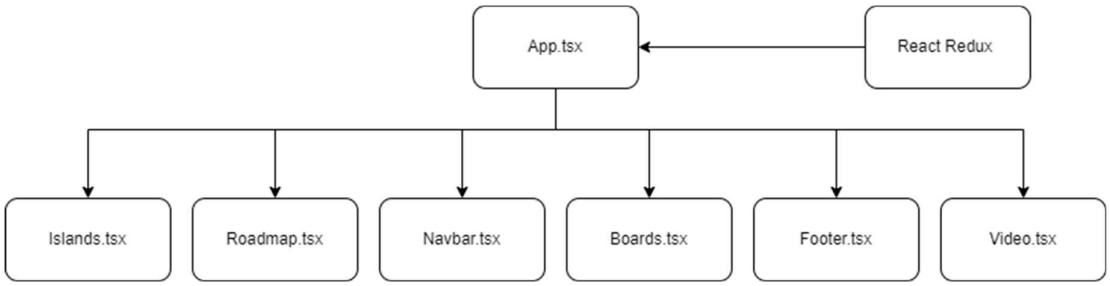
9.2.3 Struktura stránky (React tree)

Struktura react projektu je velmi důležité téma i proto, aby bylo možné porozumět struktuře stránky a hierarchii komponent. Tato stránka je navržena tak, aby všechny komponenty byly

schopné komunikovat mezi sebou a struktura projektu nebyla zbytečně složitá z důvodu budoucího testování.

Aplikace je navržena jako jednostránková aplikace, z důvodu přehlednosti a plynulosti uživatelského zážitku. Projekt je rozdělen do několika komponent, které mezi sebou mohou komunikovat tak, jak je zobrazeno na Obrázek 49. Do projektu je také zaveden ReactRedux, který zde slouží pro uchování globálních stavů. Tyto stavy jsou poté následně využity například pro REQ7 Funkce na načtení dat z peněženky NFT a následné generování obrázků.

V aplikaci se také nachází App.tsx, který slouží jako kořenová komponenta, do kterého jsou poskládané právě ostatní komponenty. Mezi tyto komponenty se řadí Islands.tsx, Roadmap.tsx, Navbar.tsx, Boards.tsx, Footer.tsx a Video.tsx. Všechny tyto komponenty jsou podrobněji popsané v kapitole implementace.



Obrázek 49 Struktura informační stránky

Je také důležité zmínit, že uspořádaní komponent probíhalo takovým způsobem, aby byla struktura co nejvíce pochopitelná a nekomplikovala následné rozšiřování, testování a úpravy projektu.

9.3 Implementace

Kapitola je zaměřená na popis implementace jednotlivých komponent, důležitých funkcí a základní přehled o tom, jak je implementována celá aplikace. Nachází se zde i výčet jednotlivých knihoven, které jsou pro tento projekt využité včetně odůvodnění proč byly vybrány.

Pro implementaci aplikace, byl zvolen jazyk Typescript a to z důvodu většího zabezpečení a silného typování, které nabízí. Je také využit ESLint. Je možné nalézt více v kapitole zabezpečení webů.

V projektu se nachází různé závislosti, které jsou v souladu s verzí Reactu 18.2.0. Jedná se například o knihovnu React-redux verze 8.0.5, která je zde využita pro držení globálních informací o přihlášeném uživateli pomocí Metamasku a informacích týkající se blockchainu. Dále se zde také nachází knihovna gsap verze 3.11.4, která je zde využita pro plynulejší animace právě, jako jsou přechody mezi pozadími a onscroll animacemi. Dále je zde využita knihovna Husky verze 4, která slouží k vytvoření tzv githooks, které zajišťují bezpečnost verzování. Také zde najdeme knihovnu styled-components verze 5.3.6, která zde slouží k veškerému stylování elementů, poskytuje nejen přehlednost, ale také efektivní znovupoužití jednotlivých stylů. Poslední a důležitou knihovnou je web3 verze 1.8.2, která nám slouží, jak k přihlášení uživatele pomocí Metamasky, tak k získávání dat z blockchainu.

9.3.1 App.tsx

Tato komponenta je implementován jako kořenová a je využita primárně pro vytvoření struktury projektu. Nachází se v něm stavy, které určují, jaké pozadí se má vyrenderovat. Dále je také vidět, že skládá všechny komponenty dohromady a tím utváří celý projekt. Obrázek 50 ukazuje, jak jsou komponenty poskládané v kořenu projektu.

```
return (
  // eslint-disable-next-line react/jsx-no-useless-fragment
  <>
    {bottom ? (
      <>
        <Navbar />
        <Background2 ref={ref} onScroll={handleScrollTop}>
          <BackgroundContainer2>
          <img style={{position: "absolute",width: "15%",left: "70%",top: "75%"}} src={data.imageTokens[3]!== undefined ? data.imageTokens[0] : "/images/Assets/Drake2.png"} alt="Dragon"/>
          <img style={{position: "absolute",width: "15%",left: "20%",top: "15%"}} src={data.imageTokens[4]!== undefined ? data.imageTokens[0] : "/images/Assets/Drake2.png"} alt="Dragon"/>
            <Boards />
            <Video title="Gameplay" src="https://www.youtube.com/embed/03Zlw7Y9qMo" />
            <img style={{position: "absolute",width: "8%",left: "70%",top: "255%"}} src={data.imageTokens[5]!== undefined ? data.imageTokens[0] : "/images/Assets/animal.png"} alt="Animal"/>
            <img style={{position: "absolute",width: "8%",left: "20%",top: "365%"}} src={data.imageTokens[6]!== undefined ? data.imageTokens[0] : "/images/Assets/animal.png"} alt="Animal"/>
            <RoadMap isAnimating={isAnimating} />
            <Footer />
          </BackgroundContainer2>
        </Background2>
      </>
    ) : (
      <>
        <Navbar />
        <Background ref={ref} onScroll={handleScroll}>
          <BackgroundContainer>
            <img style={{position: "absolute",width: "9%",left: "15%",top: "70%"}} src={data.imageTokens[0]!== undefined ? data.imageTokens[0] : "/images/Assets/Drake.png"} alt="Dragon"/>
            <img style={{position: "absolute",width: "9%",left: "80%"}} src={data.imageTokens[1]!== undefined ? data.imageTokens[0] : "/images/Assets/Drake.png"} alt="Dragon"/>
            <img style={{position: "absolute",width: "15%",left: "50%",top: "130%"}} src={data.imageTokens[2]!== undefined ? data.imageTokens[0] : "/images/Assets/Drake2.png"} alt="Dragon"/>
            <Video title="Developers info" src="https://www.youtube.com/embed/dQw4w9WgXcQ"/>
            <Islands />
          </BackgroundContainer>
        </Background>
      </>
    )}
  </>
);
}
```

Obrázek 50 App.tsx

Implementuje také logiku, která se stará o animování pozadí a přechody mezi nimi. Tato logika je zajištěna několika lokálními stavy, které zajištují, jak a kdy se má zobrazit jaké

pozadí. Všechny tyto stavy jsou závislé na pozici uživatele na stránce, i z toho důvodu jsou zde implementovány obsluhy událostí pro scroll. Obrázek 51 ukazuje implementaci událostí.

```
const handleScroll = () => {
  if (ref.current) {
    const { scrollTop, clientHeight, scrollHeight } = ref.current;
    if (scrollTop + clientHeight === scrollHeight) {
      setTimeout(() => {
        setBottom(true);
        if (ref.current !== null) {
          ref.current.scrollTo(0, 20);
        }
      }, 20);
    }
  }
};

const handleScrollTop = () => {
  if (ref.current) {
    // eslint-disable-next-line no-unused-vars
    const { scrollTop, clientHeight, scrollHeight } = ref.current;
    if (scrollTop === 0) {
      setTimeout(() => {
        setBottom(false);
        setFromBackground2(true);
      }, 20);
    }

    if (scrollTop > 1640) {
      setIsAnimating(true);
    }
  }
};
```

Obrázek 51 Obsluha scroll událostí

Také zde implementuje před načítání velkého obrázku, jako je právě pouštní pozadí. Toho je docíleno pomocí useEffect, který je vykonán pouze jednou, a to při prvním načtení App.tsx. Díky tomuto před načtení je poté aplikace plynulejší a přechod mezi pozadím má minimální odezvu. Obrázek 52 ukazuje implementaci kódu.

```
useEffect(() => {
  const preloadedImage = new Image();
  preloadedImage.src = "/images/background/Poust8k.webp";
}, []);
```

Obrázek 52 Před načítání pozadí

9.3.2 Islands.tsx

Tato komponenta není nijak zvláštní a implementuje grafické zobrazení ostrovů, které mají za úkol informovat uživatele o nabízených možnostech, tyto ostrovy jsou animovány pomocí

gsap knihovny, které vytváří animaci vznášení. Toho je docíleno pomocí useEffectu, který lze vidět Obrázek 53.

```
useEffect(() => {
  gsap.to(island1Ref.current, {
    y: 50,
    x: 20,
    yoyo: true,
    repeat: -1,
    duration: 3,
  });
  gsap.to(island2Ref.current, { y: 30, yoyo: true, repeat: -1, duration: 3 });
  gsap.to(island3Ref.current, {
    y: 20,
    x: -10,
    yoyo: true,
    repeat: -1,
    duration: 3,
  });
}, []);
```

Obrázek 53 Létající ostrovy kód

9.3.3 Navbar.tsx

Tato React komponenta, zůstává zobrazena po celou dobu aplikace, a obsluhuje většinu navigačních elementů, obsluhuje také tlačítko na přihlášení Metamasku a to pomocí zavolán metody z react redux konkrétně akce connect, tahle akce je popsaná na Obrázek 54.

```
export const connect = () => async (dispatch: Dispatch) => {
  dispatch(connectRequest());
  // Fetch the ABI for the smart contract
  const abiResponse = await fetch("/config/abi.json", {
    headers: {
      "Content-Type": "application/json",
      Accept: "application/json",
    },
  });
  const abi = await abiResponse.json();
  // Get the current network configuration
  const CONFIG = config;
  // Check if the Ethereum browser extension (e.g. MetaMask) is installed
  const { ethereum } = window as any;
  const web3 = new Web3();
  const metamaskIsInstalled = ethereum && ethereum.isMetaMask;

  if (metamaskIsInstalled) {
    // Set the provider for the Web3 object
    web3.setProvider(ethereum);

    // Get the current gas price
    const gasPrice = await web3.eth.getGasPrice();

    try {
      // Request the user's accounts
      const accounts = await ethereum.request({
        method: "eth_requestAccounts",
      });
      // Get the current network ID
      const networkId = await ethereum.request({
        method: "net_version",
      });

      // Check if the user is on the correct network
      // eslint-disable-next-line eqeqeq
      if (networkId == CONFIG.NETWORK.ID) {
        // Create a new contract object using the ABI and contract address
        const SmartContractObj = new web3.eth.Contract(abi, CONFIG.CONTRACT_ADDRESS);
        // Dispatch a success action to the store
        dispatch(
          connectSuccess({
            account: accounts[0],
            smartContract: SmartContractObj,
            web3,
            gasPrice,
          })
        );

        dispatch(updateAccount(accounts[0]));
        // Add listeners to handle account and chain changes
        ethereum.on("accountsChanged", () => {
          dispatch(updateAccount(accounts[0]));
        });
        ethereum.on("chainChanged", () => {
          window.location.reload();
        });
      } else {
        // Dispatch a failed action if the user is not on the correct network
        dispatch(connectFailed(`Change network to ${CONFIG.NETWORK.NAME}`));
      }
    } catch (err) {
      // Dispatch a failed action if an error occurs
      dispatch(connectFailed("Something went wrong."));
    }
  } else {
    // Dispatch a failed action if the Ethereum browser extension is not installed
    dispatch(
      connectFailed(
        "Install Metamask extension. If you're on mobile, please use Metamask app brows
      )
    );
  }
};
```

Obrázek 54 Akce Connect

9.3.4 RoadMap.tsx

Tato komponenta je specifická v tom, že využívá postupného načítání obrázků a tím vytváří animovanou mapu, která ukazuje jednotlivé cíle. Je tomu docíleno pomocí knihovny gsap a

postupného načítání pole elementů. Obrázek 55 ukazuje postupné načítání elementů pomocí knihovny gsap.

```
const animateImage = (imageRef: any, index: any) => {
  gsap.to(imageRef, { opacity: 1, duration: 1, delay: index * 0.1 });
};
```

Obrázek 55 Gsap animace mapy

9.3.5 Video.tsx

Jedná se o velmi jednoduchou komponentu, která jen implementuje video a jeho atributy pro zobrazení uživatelem. Celou implementaci je možné vidět na Obrázek 56.

```
function Video({title,src}: VideoProps) {
  return (
    // eslint-disable-next-line react/jsx-no-useless-fragment
    <VideoWrapper>
      <StyledSpan>{title}</StyledSpan>
      <VideoBorder>
        <VideoContainer>
          <StyledIframe
            title="Video"
            src={src}
            allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"
            allowFullScreen
          />
        </VideoContainer>
      </VideoBorder>
    </VideoWrapper>
  );
}
```

Obrázek 56 Video implementace

9.3.6 Boards

Tato komponenta, vychází z návrhu aplikace a je zde implementace textových tabulí, které slouží pro poskytnutí informací uživatelům. Implementaci komponenty zobrazuje Obrázek 57.

```
function Boards() {
  return (
    <Container>
      <StyledSpan>Play and Earn</StyledSpan>
      <BoardContainerLeft>
        <InsideBoardContainer>
          <BoardText>
            Lorem Ipsum is simply dummy text of the printing and typesetting
            industry. Lorem Ipsum has been the standard dummy text ever since
            the 1500s, when an unknown printer Lorem Ipsum is simply dummy text
            of the printing and typesetting industry. Lorem Ipsum has been the
            standard dummy text ever since the 1500s, when an unknown printer
            Lorem Ipsum is simply dummy text of the printing and typesetting
            industry. Lorem Ipsum has been the standard dummy text ever since
            the 1500s, when an unknown printer
          </BoardText>
        </InsideBoardContainer>
      </BoardContainerLeft>
      <BoardContainerRight>
        <InsideBoardContainer>
          <BoardText>
            Lorem Ipsum is simply dummy text of the printing and typesetting
            industry. Lorem Ipsum has been the standard dummy text ever since
            the 1500s, when an unknown printer Lorem Ipsum is simply dummy text
            of the printing and typesetting industry. Lorem Ipsum has been the
            standard dummy text ever since the 1500s, when an unknown printer
            Lorem Ipsum is simply dummy text of the printing and typesetting
            industry. Lorem Ipsum has been the standard dummy text ever since
            the 1500s, when an unknown printer
          </BoardText>
        </InsideBoardContainer>
      </BoardContainerRight>
    </Container>
  );
}
```

Obrázek 57 Boards.tsx

9.3.7 ReactRedux

Tato komponenta je velmi složitá a obsluhuje většinovou logiku aplikace, a to ať už se týká přihlašování Metamaskem, anebo třeba prohledávání blockchainu a získávání dat z chytrých smluv. Je rozdělen do dvou velkých částí, a to do částí blockchain a data.

Část blockchain obsahuje úvodní stav, který ukládá spousty užitečných dat, které jsou potřebné pro další fungování aplikace. Stejně také tak vytváří několik akcí, které obsluhují tyto stavy. Obrázek 58 ukazuje fungování tohoto reduxu pro blockchain část.

```
import { Action, InitialState } from "../actions";
import ActionType from "../action-types";

const initialState: InitialState = {
  loading: false,
  account: null,
  smartContract: null,
  gasPrice: "0",
  web3: null,
  errorMsg: "",
};

const blockchainReducer = (
    // eslint-disable-next-line default-param-last
    state: InitialState = initialState,
    action: Action
) => {
  switch (action.type) {
    case ActionType.CONNECTION_REQUEST:
      return {
        ...initialState,
        loading: true,
      };
    case ActionType.CONNECTION_SUCCESS:
      return {
        ...state,
        loading: false,
        account: action.payload.account,
        smartContract: action.payload.smartContract,
        gasPrice: action.payload.gasPrice,
        web3: action.payload.web3,
      };
    case ActionType.CONNECTION_FAILED:
      return {
        ...initialState,
        loading: false,
        errorMsg: action.payload,
      };
    case ActionType.UPDATE_ACCOUNT:
      return {
        ...state,
        account: action.payload.account,
      };
    default:
      return state;
  }
};

export default blockchainReducer;

```

Obrázek 58 Blockchain Reducer

Další část obsahuje stav dat, které jsou brána z chytrých smluv. Stejně tak jako jiný reducer, vytváří akce, které tyto stavy naplňují. Všechny tyto akce se poté volají skrz celý program v místech, kde je to vyžadováno. Obrázek 59 ukazuje, jak právě takový reducer pro blockchainová data vypadá.

```
5   const initialState: InitialState = {
6     loading: false,
7     imageTokens: [],
8     error: false,
9     errorMsg: "",
10  };
11
12  const dataReducer = (
13    // eslint-disable-next-line default-param-last
14    state: InitialState = initialState,
15    action: Action
16  ) => {
17    switch (action.type) {
18      case ActionType.CHECK_DATA_REQUEST:
19        return {
20          ...state,
21          loading: true,
22          error: false,
23          errorMsg: "",
24        };
25      case ActionType.CHECK_DATA_SUCCESS:
26        return {
27          ...state,
28          loading: false,
29          imageTokens: action.payload,
30          error: false,
31          errorMsg: "",
32        };
33      case ActionType.CHECK_DATA_FAILED:
34        return {
35          ...initialState,
36          loading: false,
37          error: true,
38          errorMsg: action.payload,
39        };
40      default:
41        return state;
42    }
43  };
44
45  export default dataReducer;
46
```

Obrázek 59 Data reducer

10 ZABEZPEČENÍ WEBOVÝCH ČÁSTÍ

V dnešní době je zabezpečení webových aplikací klíčovým aspektem jejich úspěchu a důvěryhodnosti. Tato kapitola je zaměřená na zabezpečení webové aplikace, která využívá TypeScript, ESLint, a je chráněna proti Cross-Site Scripting (XSS) útokům. Dále je zde využit https certifikát, který vydává Firebase. Najdeme zde také zaměření na interakci uživatelů s aplikací pomocí MetaMasku. Všechny tyto zmíněné faktory vytváří bezpečnou aplikaci.

10.1 Typescript

Typescript je klíčový z hlediska bezpečnosti naší aplikace, protože poskytuje silnou typovou kontrolu, která umožňuje předem odhalovat chyby v kódu. Toto je zásadní rozdíl oproti běžně používanému Javascriptu, který může zanést spoustu chyb do aplikace. Díky typové kontrole lze minimalizovat množství chyb a urychlit vývoj aplikace. Typescript navíc podporuje kontrolu typů v době běhu aplikace, což pomáhá snižovat počet chyb, které se objevují až v době běhu aplikace. Kromě toho Typescript zvyšuje čitelnost a udržitelnost kódu, což zlepšuje bezpečnost aplikace. Ve výsledku nám použití Typescriptu verze 4.9.4 a vyšší přináší zvýšenou bezpečnost a snižuje riziko vzniku chyb nebo zneužití bezpečnostních slabin.

10.2 ESLint

ESLint je pro tuhle aplikaci klíčový z hlediska bezpečnosti, protože poskytuje pokročilou kontrolu kódu a zajišťuje dodržování standardů a konvencí. Je využit ESLint ve verzi 5.46.1 a vyšší, který umožňuje odhalovat chyby, nekonzistence a potenciální bezpečnostní problémy v aplikačním kódu.

Tato kontrola kvality kódu pomáhá minimalizovat množství chyb a zlepšovat bezpečnost aplikace. Díky ESLintu lze zlepšit čitelnost a udržitelnost kódu, což má pozitivní vliv na celkovou bezpečnost aplikace.

ESLint také podporuje širokou škálu pravidel a pluginů, které umožňují přizpůsobit kontrolu kódu našim potřebám a zajistit, že aplikace splňuje nejvyšší bezpečnostní standardy. Díky tomu lze snižovat riziko vzniku chyb nebo zneužití bezpečnostních slabin.

Ve výsledku použití ESLintu verze 5.46.1 a vyšší přináší zvýšenou bezpečnost, lepší kontrolu kódu a snižuje riziko vzniku chyb nebo zneužití bezpečnostních slabin.

10.3 ESLint a Typescript

ESLint v kombinaci s Typescriptem byl klíčový z hlediska bezpečnosti webové aplikace, protože nám poskytuje pokročilou kontrolu kódu a zajišťuje dodržování standardů a konvencí. Byl využit ESLint ve verzi 5.46.1 a vyšší, který nám umožnil odhalovat chyby, nekonzistence a potenciální bezpečnostní problémy v kódu.

Díky integraci s Typescriptem bylo možné kombinovat výhody silné typové kontroly s kontrolou kvality kódu, což nám pomohlo minimalizovat množství chyb a zlepšit bezpečnost aplikace. Tato kombinace také umožnila zlepšit čitelnost a udržitelnost kódu, což má pozitivní vliv na celkovou bezpečnost naší aplikace.

Ve výsledku použití ESLintu v kombinaci s Typescriptem verze 4.9.4 a vyšší přináší zvýšenou bezpečnost, lepší kontrolu kódu a snižuje riziko vzniku chyb nebo zneužití bezpečnostních slabin.

10.4 Cross-site Scripting

Cross-site scripting (XSS) je typ bezpečnostního útoku, při kterém útočník vkládá škodlivý kód do webových stránek, který je následně spuštěn v prohlížeči oběti. Tento útok může vést ke kompromitaci citlivých dat, jako jsou přihlašovací údaje, osobní informace nebo session cookies. Předcházení XSS útokům je nezbytné pro zajištění bezpečnosti webových aplikací a ochranu uživatelů.

Pro ověření, toho, zda webové aplikace neobsahují bezpečnostní chyby spojené s XSS, byl využit známý nástroj XSS scanner, který lze najít na adrese [66]. Tento nástroj provádí automatické testy na naší webové stránce a detekuje možné zranitelnosti spojené s XSS útoky.

Ačkoli tento nástroj může poskytnout důležitou kontrolu a pomoci nám odhalit některé chyby, je důležité si uvědomit, že žádný nástroj není 100% spolehlivý a nemůže zaručit naprostou bezpečnost. Pro zajištění co nejvyšší úrovně ochrany proti XSS útokům byly tedy provedeny i další bezpečností kroky, a to například ochrana aplikace právě pomocí typescriptu a ESLint.

V našem případě však výsledky testu z online nástroje pentest-tools, neodhalil žádné závažné chyby. Je však nutno brát v potaz, že nebyla využitá plnohodnotná verze, ale pouze ta, kterou

nabízí služba zdarma, a tak je nutno nedávat tomuto testu 100 % hodnotu, oba tyto reporty lze najít v přílohách pod názvem LandingPageReport a MintingPageReport.

I přes všechna tato zabezpečení je nutno, dále aplikace aktualizovat a kontrolovat aplikaci, aby se předešlo novým neodhaleným bezpečnostním hrozbám, které mohou vznikat při dalším vývoji aplikace.

10.5 Https (firebase)

HTTPS (Hypertext Transfer Protocol Secure) je zabezpečená verze HTTP protokolu, která chrání data přenášená mezi webovým prohlížečem a serverem prostřednictvím šifrování. Toto zabezpečení je zásadní pro ochranu uživatelů při používání webových aplikací, neboť brání odposlechu a útokům typu man-in-the-middle. Bezpečnost HTTPS je založena na SSL/TLS certifikátech, které zajišťují šifrování a ověřování totožnosti serveru.

V aktuální fázi projektu byl použit testovací hosting na Firebase, což je platforma od společnosti Google určená pro rychlý a bezpečný vývoj webových aplikací. Díky tomuto řešení nebylo nutné se zabývat HTTPS certifikátem, protože Firebase ho automaticky poskytuje. To znamená, že naše aplikace je chráněna šifrováním bez nutnosti řešit správu certifikátu.

Nicméně, tato situace představuje "černou skříňku", kterou můžeme použít pouze v testovací fázi. V budoucnu je v plánu prověřit, vybrat a spravovat vlastní bezpečnostní certifikát. I když HTTPS certifikát poskytovaný v této testovací fázi nemusí být vždy 100% důvěryhodný, stále nám poskytuje kvalitní, rychlé a snadno škálovatelné řešení hostingu bez nutnosti platit třetím stranám za tyto služby.

10.6 Metamask

Metamask je rozšíření pro webové prohlížeče (jako Chrome, Firefox, Brave a další), které umožňuje uživatelům přistupovat k decentralizovaným aplikacím (dApps) postaveným na Ethereum blockchainu.

Metamask v aplikaci slouží jako peněženka pro Ethereum blockchainy a ERC-20 tokeny, což umožňuje uživatelům spravovat své kryptoměny a provádět transakce. Metamask poskytuje klíčovou vrstvu zabezpečení mezi uživatelem a naší aplikací. Uživatelům umožňuje spravovat své soukromé klíče, které jsou šifrovány a uloženy v rámci prohlížeče. Metamask také funguje jako prostředník mezi uživatelem a blockchainem, protože

zpracovává a podepisuje transakce, aniž by soukromé klíče opustily uživatelovo zařízení. Tímto způsobem Metamask pomáhá chránit uživatele před útoky, jako je phishing nebo keylogging.

Díky všem těmto aspektům, aplikace nepotřebuje dodatečné zabezpečení, protože uživatel se přihlašuje pomocí Metamasku, a ten zastupuje jeho uživatelský účet, právě i z tohoto důvodu není nutno řešit GDPR, protože adresa peněženky se nebere jako soukromý údaj.

Dodatečné zabezpečení, které může být integrováno je upozorňovat uživatele na to, aby si zálohoval své privátní klíče a ukládal je na bezpečném místě. Popřípadě využíval hardwarové peněženky.

11 UNITY ASSETS PACKAGE

Tato část diplomové práce se zabývá vytvořením balíčků Unity Asset. V této praktické části je kladen důraz na vytvoření jednotlivých skriptů a scén. Vzhledem k tomu, že se jedná pouze o doplňkové balíčky, které mají sloužit jako pomoc při vývoji testovací aplikace, byl kladen hlavní důraz na funkčnost jednotlivých částí.

V rámci projektu jsou k dispozici dva Unity Asset balíčky a to MovementPackage a UIPackage. Jeden z nich je zaměřen na uživatelské rozhraní, jako je menu, inventář a nastavení. Druhý balíček se zaměřuje na pohyb hráče po mapě pomocí GPS. Z tohoto důvodu bylo nutné zajistit také mapu, po které se model bude pohybovat.

11.1 UI package

Tento balíček byl vytvořen za účelem poskytnutí uživatelského rozhraní (UI) pro aplikaci. Je rozdělen do několika scén, přičemž každou scénu ovládají skripty, které jsou přiřazeny odpovídajícím prvkům typu "canvas". V balíčku se nachází šest prvků typu "canvas", pět herních objektů, šest skriptů, dvakrát objekt typu prefab a několik obrázků. Tyto obrázky jsou buď generovány umělou inteligencí, nebo jsou převzaty z volně dostupných zdrojů, které nezakazují jejich použití z důvodu licenčních omezení.

11.1.1 Canvas-MainScreen

Tento prvek slouží k zobrazení hlavního menu, které umožňuje uživateli ovládat hru a přistupovat k ostatním prvkům. Je navržen tak, aby byl plně responsivní a přizpůsoboval se velikosti zařízení.

Tento prvek se nazývá „Canvas-MainScreen" a využívá funkci "OpenMenu", která je implementována ve scriptu "GameManager". Tato funkce slouží k obsluze tlačítka a má za úkol skrýt "Canvas-MainScreen" a zobrazit "Canvas-MenuOptions".

11.1.2 Canvas-MenuOptions

Jedná se o velmi obsáhlý prvek typu "canvas", který ovládá logiku celého menu. Obsahuje několik prvků, které umožňují uživateli navigovat v aplikaci. Tento prvek "canvas" lze rozdělit na dvě rozsáhlé části – horní a dolní menu.

V dolním menu se nachází několik prvků. Prvním z nich je tlačítko "ExitMenuButton", které slouží k opuštění "MenuOptions". Toto tlačítko volá funkci "CloseMenu" ve scriptu "GameManager".

Dalším prvkem je "OpenOwnMonsters", které volá funkci "OpenOwnMonsterMenu". Tato funkce zatím nemá žádnou konkrétní funkčnost a je připravena pro budoucí vývoj.

Dalším prvkem v tomto "canvas" je "OpenAllExistingMonsters", které volá funkci "OpenAllExistingMonstersMenu". Tato funkce zajišťuje zavření aktuálního menu a zobrazení "canvas" s názvem "MonsterBook". Následně je zavolána funkce "DisplayCards", která slouží k naplnění "DisplayCards".

Dalším prvkem je "OpenShop", který slouží pouze jako placeholder a je s ním spojena funkce "OpenShopMenu", která zatím nevykonává žádnou činnost.

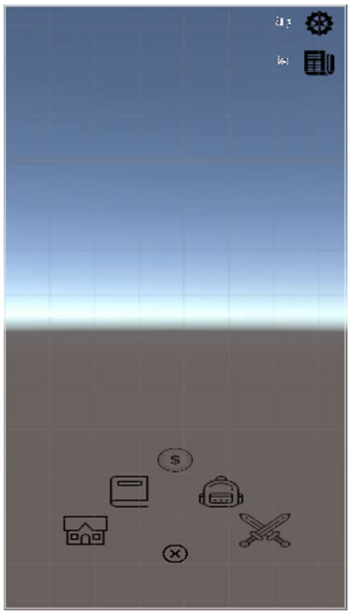
Dalším prvkem je "OpenInventory", který volá funkci "OpenInventoryMenu". Tato funkce skryje aktuální "canvas" s názvem "MenuOptions" a zobrazí "canvas" s názvem "Inventory". Následně je zavolána funkce "DisplayItemCards", která slouží k naplnění inventáře.

Posledním prvkem v dolní části menu je "OpenBattle", který rovněž slouží jako placeholder a je s ním spojena funkce "OpenBattleMenu".

V horním menu se nachází prvek "OpenSettingButton", který je propojen s funkcí "OpenSettingMenu". Tato funkce skryje aktuální "canvas" a zobrazí nový "canvas" s názvem "Setting".

Druhým prvkem v horním menu je "OpenNewsButton", který volá funkci "OpenNewsMenu". Tato funkce skryje aktuální "canvas" a zobrazí "canvas" s novinkami, který je nazván "Canvas-News".

Grafické zobrazení je možné pozorovat na Obrázek 60.

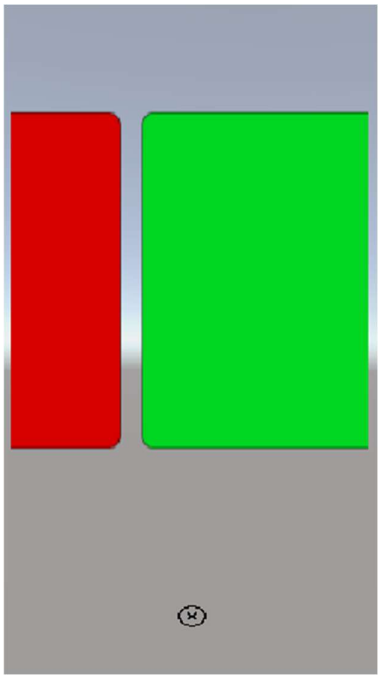


Obrázek 60 Canvas-MenuOptions

11.1.3 Canvas-News

Jedná se o prvek typu canvas, který má za úkol poskytnout uživateli informace a novinky ohledně hry. Tento prvek obsahuje "ExitMenuButton", které slouží k opuštění tohoto menu. Po stisknutí tohoto tlačítka se zavolá funkce z "GameManageru" s názvem "CloseNewsMenu", která skryje aktuální canvas a znovu zobrazí "menuOptions" canvas.

Dále v tomto prvku se nachází prvek "Scroll", který obsahuje tři hlavní karty, mezi kterými lze pomocí tohoto prvku scrollovat. Náhled tohoto prvku lze vidět na Obrázek 61.



Obrázek 61 Canvas-News

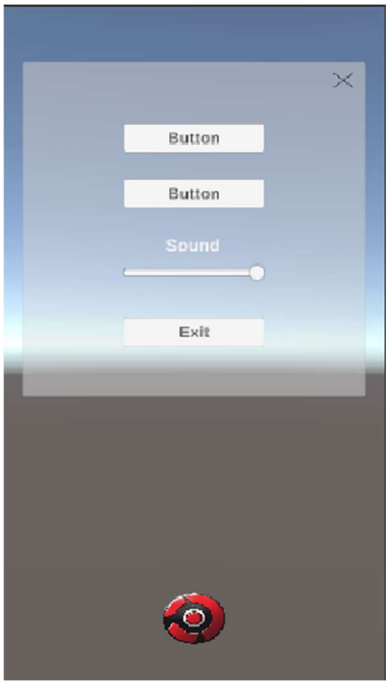
11.1.4 Canvas-Setting

Jedná se o prvek typu canvas, který má za cíl poskytnout uživateli hlavní menu pro nastavení hry. Zde se nachází tlačítko "ExitMenuButton", které je propojeno s funkcí "CloseSettingMenu". Tato funkce zavírá aktuální menu canvas a otevírá "canvas-MenuOptions". Dále zde jsou také tlačítka "ButtonOperation1" a "ButtonOperation2", která jsou připravena pro budoucí využití a mají příslušné funkce připojené.

Dalším důležitým prvkem je "SliderSound", který slouží k nastavení zvuku ve hře. K tomuto slideru je připojen skript, který je propojen s "MainMixerem". Ten je poté ovládán funkcí "SetVolume", která při posouvání slideru dynamicky nastavuje aktuální hodnotu do audioMixeru.

Posledním prvkem, který se zde nachází, je tlačítko "ButtonExit". Na něj je navázána funkce "QuitGame", která ukončuje aplikaci. Všechny tyto funkce se, na rozdíl od ostatních, nachází ve skriptu "SettingMenu".

Obrázek 62 ukazuje, jak tento element vypadá.



Obrázek 62 Canvas-Setting

11.1.5 Canvas-MonsterBook

Jedná se o prvek typu canvas, který slouží k zobrazení všech existujících monster ve hře. Tato monstra jsou zobrazována pomocí karet, které jsou vytvořené jako prefaby. V tomto prvku je také použit prvek scroll, který umožňuje uživateli procházet více karet. Stejně jako v každém našem canvasovém prvku se zde nachází tlačítko "ExitMenuButton", které je propojeno s funkcí "CloseAllExistingMonstersMenu", která zavírá aktuální canvas a otevírá "canvas-MenuOptions".

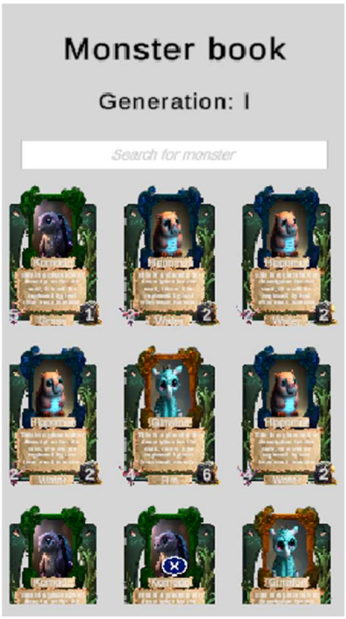
Dále zde najdeme dva textové pole, které slouží jako navigace pro uživatele. Zde se také nachází "SearchBar", který v současné době nemá funkčnost, ale v budoucnu bude sloužit pro rychlé vyhledávání v kolekci.

Posledním důležitým prvkem je "CardPanel", který obsahuje všechny naše karty. Obsah těchto karet je generován dynamicky. Každá karta se skládá z elementů jako je "Card-Frame-mask", který slouží k uložení obrázku monstra, a "CardFrame", který zobrazuje rámeček, ve kterém je umístěna fotka monstra. Dále se zde nachází "CardDescription-Image", který zobrazuje obrázek a textový popis monstra, a "Type-image" element, který obsahuje obrázek

a textový popis typu monstra. Dalšími prvky na kartě jsou "Health-Image", "Attack-Image" a "Name-Image", které zobrazují konkrétní statistiky monstra.

Všechny tyto karty jsou ovládány pomocí skriptu "CardManager". Tento skript obsahuje veřejnou kolekci karet typu "Card". Typ "Card" je rovněž skript, který slouží pouze k definici dat pro naši kolekci. Skripty pak pomocí funkcí "DisplayCards" získávají data z "CardManageru" a dynamicky naplňují jednotlivé karty. Zajímavostí je, že barva karty se mění v závislosti na jejím typu.

Ukázku "Canvas-MonsterBook" lze sledovat na Obrázek 63.



Obrázek 63 Canvas-MonsterBook

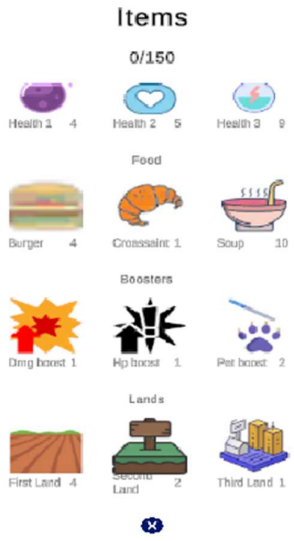
11.1.6 Canvas-Inventory

Jedná se o prvek typu canvas, který slouží k zobrazení inventáře. Inventář je zobrazen jako série skupin předmětů, mezi kterými lze scrollovat. V tomto prvku se nachází tlačítko "ExitMenuButton", které je propojeno s funkcí "CloseInventoryMenu", která skryje aktuální canvas a zobrazí "Canvas-MenuOptions". Dále zde najdeme dva textové prvky - "Inventory-Text" a "Inventory-Number", které slouží k zobrazení základních informací.

Nejdůležitější částí tohoto prvku je prvek "Scroll", který obsahuje "CardPanel", do kterého jsou umístěny jednotlivé prefabrikované prvky "InventoryCardBody". Tyto prvky

uchovávají informace o jednotlivých předmětech. Každá tato prefabrikovaná karta se skládá z "InventoryName", které slouží k pojmenování příslušné skupiny předmětů, a také z "ItemGridu", ve kterém jsou umístěny 3 předměty dané skupiny. Každý předmět je reprezentován "ItemBody", který obsahuje název, počet a obrázek. Proces naplňování těchto inventářových karet je zajištěn pomocí skriptu "InventoryCardManager", který využívá podobnou techniku jako zmíněný "CardManager". Tento skript je propojen s "InventoryCardManagerem", který vytváří veřejnou kolekci typu "InventoryCard". Tato kolekce je následně naplněna pomocí funkce "DisplayItemsCards", která získává informace z "InventoryCardManageru" a dynamicky generuje předměty při spuštění hry.

Inventář je možné vidět na Obrázek 64.



Obrázek 64 Canvas-Inventory

11.2 Movement package

Tento balíček byl vytvořen s cílem poskytnout uživatelům pohyb po mapě pomocí GPS souřadnic. Jedná se o klíčovou část aplikace, která ovlivňuje další vývoj. Balíček je rozdělen na herní objekt "LocationBasedGame", který obsahuje herní objekt "Map" pro vytváření a správu mapy v aplikaci, a "LocationProvider" pro zajištění pohybu po mapě pomocí GPS, a to jak na fyzickém zařízení, tak v editoru pro testování. Dále je zde také herní postava, na kterou jsou navázány skripty zajišťující její pohyb.

Pro vytvoření tohoto balíčku je využíván balíček a služba poskytovaná společností Mapbox [68]. Díky této službě jsme schopni efektivně vyvíjet naši aplikaci, protože nám poskytuje potřebné skripty pro ovládání postavy a také samotnou mapu, kterou lze upravovat v jejich konfigurátoru pomocí vrstev.

11.2.1 Map

Pro vytvoření mapy je nejprve nutné importovat zmíněný balíček od Mapboxu, konkrétně část MapMovement. Následně je potřeba vyplnit a získat API klíče, které je možné získat na jejich stránce. Je třeba si uvědomit, že tyto API klíče jsou placené a jejich cena se odvíjí od dynamického využití. Z tohoto důvodu je nutné tyto API klíče bezpečně ukládat.

Na stránce byla vytvořena mapa typu "streets", ze které byly odstraněny všechny vrstvy, kromě "road-simple", "building", "water", "waterway", "pitch-outline" a "land". Po zachování pouze těchto vrstev jsme vytvořili mapu, kterou budeme používat pro testování naší aplikace. Tuto mapu lze vidět na Obrázek 65.

Obrázek 65 Náhled Mapy

Následně bylo pouze změněno "Data Source" objektu na vlastní typ "mapbox://styles/rabeles/clhdw6vby014o01qu4w9pfbt4". Poté byla přidána funkce "feature" typu "building" a byl upraven její polygon s výškou 2 a škálovacím faktorem 3. Tato konfigurace nám umožňuje přenést vytvořenou mapu do naší aplikace a zároveň nastavit 3D výšku budov tak, aby mapa působila co nejatraktivněji na uživatele.

Tímto způsobem jsme vytvořili kompletní mapu pro naši aplikaci, která je snadno upravitelná a zároveň optimalizovaná. Aplikace generuje pouze okolí a části mapy, ve kterých se hráč nachází, aby šetřila data a výkon zařízení.

Finální verze mapy v unity aplikaci, lze vidět na Obrázek 66.

Obrázek 66 Finální mapa v unity aplikaci

11.2.2 LocationProvider

Tento herní objekt je zase založen na balíčku od Mapboxu, stejně jako mapa. Jedná se o prefab, který obsahuje skripty pro pohyb hráče na různých zařízeních. Nejprve je potřeba tento prefab přetáhnout do sloupce MapManageru, který aktuálně používáme. Tím se nastaví tento objekt jako poskytovatel polohy.

Dále je nutné nastavit Device Location Provider Unity a Device Provider Location Android, které slouží k určování polohy hráče. Tyto skripty také zajišťují veškerá oprávnění a další potřebné nastavení pro získávání geolokačních dat. Posledním krokem je vyplnění TransformLocationProvider a EditorLocationProvider. EditorLocationProvider je určen pro ladění a ladící účely aplikace. Můžete si vybrat mezi různými skripty, které umožní buď

automatický pohyb hráče podle předdefinovaných částí mapy, nebo posunutí hráče na předem určené souřadnice.

Pomocí tohoto herního objektu od Mapboxu snadno zajišťujeme jednu z hlavních funkcí naší aplikace, a to pohyb po mapě.

11.2.3 Vodoo-Dool (Player)

Jedná se o herní objekt, který představuje postavu hráče. Pro tuto postavu byl vybrán volně dostupný model Voodoo panenky. Pro správné fungování celé aplikace je nezbytné připojit k tomuto modelu dva herní skripty, které spolu komunikují a zajišťují pohyb postavičky po naší dříve vytvořené mapě. Tyto skripty se nazývají "ImmediatePositionWithLocationProvider" a "RotateWithLocationProvider". Parametr "RotationFollowFactor" byl nastaven na hodnotu 2. První ze zmíněných skriptů zajišťuje pohyb postavy a druhý skript se stará o rotaci postavy ve směru zařízení hráče.

Náhled na herní model umístěný v naší mapě, lze pozorovat na Obrázek 67.

Obrázek 67 Náhled hráče na mapě

11.2.4 Kamera

Posledním herním objektem v tomto balíčku je kamera. Jedná se o základní kameru, ke které je připojen vlastní skript nazvaný "SmoothCameraFollow". Tento skript má za úkol sledovat

náš herní postavičku. Skript "SmoothCameraFollow" přijímá objekt, který má být sledován, a poté vypočítává aktuální pozici tohoto objektu. Kamera je pak plynule přesouvána na tuto pozici, aby následovala objekt.

ZÁVĚR

V teoretické části mé diplomové práce byla provedena zevrubná rešerše problematiky a poskytnut úvod do fungování NFT her. Byly vysvětleny základy blockchainu a EVM, včetně jejich nejznámějších standardů. Dále byla probrána problematika webového vývoje a jednotlivých WEB3 knihoven. V závěru teoretické části jsem se zaměřil na Blockchain hry, jejich rozdělení, tržní kapitalizaci a přehled těch nejznámějších.

V praktické části této práce jsem vytvořil NFT kolekci ve vhodném formátu. Po pečlivé úvaze a analýze jsem se rozhodl pro standard ERC-721 a ukládání dat přímo na blockchain. Toto rozhodnutí bylo učiněno s ohledem na decentralizaci a potenciální úspory, které tato metoda nabízí oproti tradiční databázi, kterou jsem původně plánoval použít. Přestože toto řešení představuje určité výzvy, věřím, že výhody, které poskytuje, převažují nad potenciálními nevýhodami. Implementace a deploy na testnet proběhl bez komplikací.

V další fázi jsem navrhl a realizoval databázi a backend, pro který jsem zvolil jazyk C# a .NET, zatímco pro databázi jsem upřednostnil PostgreSQL. Backend je implementován jako čtyřvrstvá architektura, která přistupuje k databázi pomocí repository patternu, na který byl také kladen největší důraz. Dále jsem vytvořil modely a návrh jednotlivých dat, které je třeba ukládat do této databáze.

Následně byly vytvořeny dvě webové aplikace – minting aplikace a landing page. V těchto fázích proběhlo také generování grafických prvků pomocí moderní technologie MidJourney. Pro vývoj samotných webových aplikací, byl zvolen jazyk Typescript a moderní Web3 knihovny pro integraci blockchainu. U obou webových aplikací bylo provedeno zabezpečení pomocí HTTPS certifikátu, využití silně typového jazyka a kontrola kódu pomocí ESLint. Dále byl proveden jednoduchý penetrační test, jehož provedení zajišťovala třetí strana, která poskytuje základní testy zdarma.

Ve finální části mé diplomové práce, jsem vytvořil dva unity asset package. První z nich se zaměřil na vytvoření UI a základní funkcionality, jako je například inventář anebo nastavení. Následně jsem vytvořil základní mapu a pohyb na této mapě pomocí GPS, k čemuž jsem využil třetí stranu, konkrétně Mapbox, který nabízí právě tato řešení na klíč. Tato služba bude v aplikaci využita i do budoucna pro své široké možnosti využití pro GO hry.

Má diplomová práce otevírá široké možnosti pro další vylepšení a mnoho z jejích částí bude využito pouze jako blueprint vzhledem k rychlému vývoji trendů v oblasti blockchainu.

Plánuji aplikaci dále rozvíjet a dotáhnout další části do úspěšného konce. Jedná se o reálný projekt, pro který budu v budoucnu hledat sponzora, aby se urychlil její vývoj. Tato komplexní aplikace i pro větší tým developerů představuje výzvu, která zabere delší dobu na vývoj.

SEZNAM POUŽITÉ LITERATURY

[1] VAN HIJFTE, Stijn. Blockchain Platforms: A Look at the Underbelly of Distributed Platforms. Howest Applied University College: Morgan & Claypool Publishers. ISBN 9781681738932. Dostupné z: doi:10.2200/S01022ED1V01Y202006CSL011.

[2] alikhan91. What is P2P (Peer-to-Peer) Network? [online]. Steemit, 2018-03-12 [cit. 2023-04-07]. Dostupné z: https://steemit.com/networking/@alikhan91/what-is-p2p-peer-to-peer-network

[3] JUDMAYER, Aljosha, Nicholas STIFTER, Katharina KROMBHOLZ a Edgar WEIPPL. Blocks and Chains Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms. Morgan & Claypool Publishers, 2017. ISBN 9781627057165. Dostupné z: doi:10.2200/S00773ED1V01Y201704SPT020.

[4] Geek Culture. Proof of Stake vs. Proof of Work in Cryptocurrencies [online]. Medium, 2021-08-04 [cit. 2023-04-07]. Dostupné z: https://medium.com/geekculture/proof-of-stake-vs-proof-of-work-in-cryptocurrencies-f558dbbaec4d

[5] Investopedia. Public, Private, and Permissioned Blockchains Compared [online]. Investopedia, 2021-03-31 [cit. 2023-04-07]. Dostupné z: https://www.investopedia.com/news/public-private-permissioned-blockchains-compared/

[6] Swan, M. On the Origins and Variations of Blockchain Technologies [online]. In: Blockchain Research Handbook [ed. by S. T. Ali and S. Clarke]. Hoboken, NJ: John Wiley & Sons, 2021, s. 1-32. ISBN 978-1-119-99055-2. Dostupné z: https://www.researchgate.net/publication/344962789_On_the_Origins_and_Variations_of_Blockchain_Technologies

[7] Narayanan, Arvind; Bonneau, Joseph; Felten, Edward; Miller, Andrew; Goldfeder, Steven (2016). Bitcoin and cryptocurrency technologies: a comprehensive introduction. Princeton: Princeton University Press. ISBN 978-0-691-17169-2.

[8] Statista. Worldwide Bitcoin blockchain size as of January 2022 [online]. [cit. 2023-04-07]. Dostupné z: https://www.statista.com/statistics/647523/worldwide-bitcoin-blockchain-size/

[9] Ethereum.org. Ethereum Virtual Machine (EVM) [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/cs/developers/docs/evm/

[10] Merriam-Webster. Definition of NFT [online]. [cit. 2023-04-07]. Dostupné z: https://www.merriam-webster.com/dictionary/NFT

[11] Ethereum.org. NFTs on Ethereum [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/en/nft/

[12] Zen of Fine Art. The Beginning of NFTs: A Brief History of NFT Art [online]. [cit. 2023-04-07]. Dostupné z: https://www.zenofineart.com/blogs/news/the-beginning-of-nfts-a-brief-history-of-nft-art

[13] Bored Ape Yacht Club [online]. [cit. 2023-04-07]. Dostupné z: https://boredapeyachtclub.com/

[14] OpenZeppelin Contracts. Different Kinds of Tokens [online]. Verze 3.x. [cit. 2023-04-07]. Dostupné z: https://docs.openzeppelin.com/contracts/3.x/tokens#different-kinds-of-tokens

[15] Cointelegraph. Fungible vs. Non-Fungible Tokens: What is the Difference? [online]. [cit. 2023-04-07]. Dostupné z: https://cointelegraph.com/learn/fungible-vs-nonfungible-tokens-what-is-the-difference

[16] Third Web. EVM Compatible Blockchains and Ethereum Virtual Machine [online]. 2021-08-26 [cit. 2023-04-07]. Dostupné z: https://blog.thirdweb.com/evm-compatible-blockchains-and-ethereum-virtual-machine/

[17] Ethereum.org. What is Ethereum? [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/en/what-is-ethereum/

[18] Binance. BNB [online]. [cit. 2023-04-07]. Dostupné z: https://www.binance.com/en/bnb

[19] Polygon. Polygon [online]. [cit. 2023-04-07]. Dostupné z: https://polygon.technology/

[20] Ethereum Developers. ERC-721 Non-Fungible Token Standard [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/en/developers/docs/standards/tokens/erc-721/

[21] Coinmonks. The Most Popular NFT Standards: Make Your Art Publicly Available in Code [online]. Medium, 2021-08-24 [cit. 2023-04-07]. Dostupné z: https://medium.com/coinmonks/the-most-popular-nft-standards-make-your-art-publicly-available-in-code-189be4d75a01

[22] OpenZeppelin Contracts. OpenZeppelin Contracts ERC721 Token Contract [online]. Verze 2.x. [cit. 2023-04-07]. Dostupné z: https://docs.openzeppelin.com/contracts/2.x/api/token/erc721

[23] OpenZeppelin Contracts. ERC721.sol [online]. [cit. 2023-04-07]. Dostupné z: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC721/ERC721.sol

[24] OpenZeppelin Contracts. OpenZeppelin Contracts ERC1155 Token Contract [online]. Verze 3.x. [cit. 2023-04-07]. Dostupné z: https://docs.openzeppelin.com/contracts/3.x/api/token/erc1155

[25] OpenZeppelin Contracts. ERC1155.sol [online]. [cit. 2023-04-07]. Dostupné z: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC1155/ERC1155.sol

[26] Ethereum Developers. ERC-20 Token Standard [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/en/developers/docs/standards/tokens/erc-20/

[27] OpenZeppelin Contracts. OpenZeppelin Contracts ERC20 Token Contract [online]. Verze 3.x. [cit. 2023-04-07]. Dostupné z: https://docs.openzeppelin.com/contracts/3.x/api/token/erc20

[28] OpenZeppelin Contracts. ERC20.sol [online]. [cit. 2023-04-07]. Dostupné z: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol

[29] GeeksforGeeks. Web Development [online]. [cit. 2023-04-07]. Dostupné z: https://www.geeksforgeeks.org/web-development/

[30] BrainStation. What is Web Development? [online]. [cit. 2023-04-07]. Dostupné z: https://brainstation.io/career-guides/what-is-web-development

[31] GeeksforGeeks. Difference Between Web Application and Website [online]. [cit. 2023-04-07]. Dostupné z: https://www.geeksforgeeks.org/difference-between-web-application-and-website/

[32] hackr.io. Best JavaScript Frameworks, Libraries and Tools to Use in 2022 [online]. [cit. 2023-04-07]. Dostupné z: https://hackr.io/blog/best-javascript-frameworks

[33] Khan, Z. Top Popular Frontend Frameworks in 2023 [online]. LinkedIn, 2022-10-17 [cit. 2023-04-07]. Dostupné z: https://www.linkedin.com/pulse/top-popular-frontend-frameworks-2023-zuraiz-khan/

[34] SANTANA ROLDAN, Carlos. React 17 Design Patterns and Best Practices: Design, build, and deploy production-ready web applications using industry-standard practices. 3rd. Packt Publishing, 2021. ISBN 978-1800560444

[35] AltexSoft. The Good and The Bad of Angular Development [online]. 2019-07-11 [cit. 2023-04-07]. Dostupné z: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/

[36] Webuters. Advantages and Disadvantages of Using Vue.js [online]. [cit. 2023-04-07]. Dostupné z: https://www.webuters.com/advantages-and-disadvantages-of-using-vuejs

[37] Hashnode. What is Web3.js? An Introduction into the Web3.js Libraries [online]. [cit. 2023-04-07]. Dostupné z: https://web3.hashnode.com/what-is-web3js-an-introduction-into-the-web3js-libraries

[38] DeveloperUpdates. Top Web 3 Frameworks and Libraries [online]. [cit. 2023-04-07]. Dostupné z: https://www.developerupdates.com/blog/top-web-3-frameworks-and-libraries

[39] Web3.js. Web3.js Documentation [online]. Verze 1.7.1. [cit. 2023-04-07]. Dostupné z: https://web3js.readthedocs.io/en/v1.7.1/

[40] Ethers.js. Ethers.js Documentation [online]. Verze 5.x. [cit. 2023-04-07]. Dostupné z: https://docs.ethers.org/v5/

[41] OpenZeppelin Contracts. OpenZeppelin Contracts Documentation [online]. Verze 4.x. [cit. 2023-04-07]. Dostupné z: https://docs.openzeppelin.com/contracts/4.x/

[42] Cloudflare. What is Web Application Security? [online]. [cit. 2023-04-07]. Dostupné z: https://www.cloudflare.com/learning/security/what-is-web-application-security/

[43] DailyRazor. 10 Web Application Security Best Practices to Secure Your Data [online]. 2021-03-26 [cit. 2023-04-07]. Dostupné z: https://www.dailyrazor.com/blog/10-web-application-security-best-practices-to-secure-your-data/

[44] PlayToEarn. What are Blockchain Games? [online]. [cit. 2023-04-07]. Dostupné z: https://www.playtoearn.online/what-are-blockchain-games/

[45] NFT Gators. What is a Blockchain Game? [online]. [cit. 2023-04-07]. Dostupné z: https://www.nftgators.com/what-is-a-blockchain-game/

[46] Cointelegraph. Blockchain Gaming Adoption Means More Options for Gamers [online]. 2021-03-04 [cit. 2023-04-07]. Dostupné z: https://cointelegraph.com/news/blockchain-gaming-adoption-means-more-options-for-gamers

[47] MarketsandMarkets. Blockchain Gaming Market by Provider, Gaming Type (Adventure & Role-playing Games, Casino Games, Sports Betting, Others), Blockchain

Type (Public, Hybrid), Deployment, Device, Region - Global Forecast to 2026 [online]. [cit. 2023-04-07]. Dostupné z: https://www.marketsandmarkets.com/Market-Reports/blockchain-gaming-market-167926225.html

[48] Cove Markets. What is Axie Infinity? [online]. [cit. 2023-04-07]. Dostupné z: https://www.covemarkets.com/what-is-axie-infinity/

[49] Axie Infinity. Axie Infinity [online]. [cit. 2023-04-07]. Dostupné z: https://axieinfinity.com/

[50] Investopedia. What Is Axie Infinity? [online]. 2021-08-24 [cit. 2023-04-07]. Dostupné z: https://www.investopedia.com/what-is-axie-infinity-5220657

[51] Decentraland. Decentraland [online]. [cit. 2023-04-07]. Dostupné z: https://decentraland.org/

[52] S. Siddiqa, N. Javaid, M. Tariq, A. Javed, M. Imran, M. Shahid. Blockchain-based multi-player games: A survey. IEEE Xplore [online]. 2021 [cit. 2023-04-07]. DOI: 10.1109/ACCESS.2021.3068461. Dostupné z: https://ieeexplore.ieee.org/document/9951366

[53] CoinMarketCap. The Sandbox (SAND) price, marketcap, chart, and fundamentals info [online]. [cit. 2023-04-07]. Dostupné z: https://coinmarketcap.com/currencies/the-sandbox/

[54] VentureBeat. The Sandbox raises $2 million more to build out blockchain-based game world [online]. 2020-05-20 [cit. 2023-04-07]. Dostupné z: https://venturebeat.com/business/the-sandbox-raises-2-million-more-to-build-out-blockchain-based-game-world/

[55] Metaverse Insider. 7 Companies With Strategic Partnerships With The Sandbox [online]. 2022-07-04 [cit. 2023-04-07]. Dostupné z: https://metaverseinsider.tech/2022/07/04/7-companies-with-strategic-partnerships-with-the-sandbox/

[56] P2P [obrázek]. In: ALIKHAN, Muhammad [online]. 2021 [cit. 2023-04-07]. Dostupné z: https://steemitimages.com/p/TZjG7hXReeVoAvXt2X6pMxYAb3q65xMju8wryWxKrsghkMwdbZa52AsJ5wtwwBFRZ2xRykPyRLY4udzzCkwpJQsRpbGFDM3sjwE6vEa8joh4W7nD9nGjxsWNPDiavKaTM2Nr9e9CqHScuC?format=match&mode=fit&width=640

[57] HASH TREE [obrázek]. In: Wikimedia Commons [online]. [cit. 2023-04-07]. Dostupné z:

https://upload.wikimedia.org/wikipedia/commons/thumb/9/95/Hash_Tree.svg/1200px-Hash_Tree.svg.png

[58] VAN HIJFTE, Stijn. Blockchain Platforms: A Look at the Underbelly of Distributed Platforms. Morgan & Claypool Publishers, 2020. ISBN 9781681738932. Obrázek na straně 55 [online]. Dostupné z: doi:10.2200/S01022ED1V01Y202006CSL011 [cit. 2023-04-07].

[59] ETHEREUM FOUNDATION. Gas [obrázek]. In: Ethereum [online]. [cit. 2023-04-07]. Dostupné z: https://ethereum.org/static/9628ab90bfd02f64cf873446cbdc6c70/302a4/gas.png

[60] YUGA LABS. Bored Ape Yacht Club [obrázek]. In: Forbes [online]. 10. září 2021 [cit. 2023-04-07]. Dostupné z: https://imageio.forbes.com/specials-images/imageserve/6170e01f8d7639b95a7f2eeb/Sotheby-s-NFT-Natively-Digital-1-2-sale-Bored-Ape-Yacht-Club--8817-by-Yuga-Labs/0x0.png?format=png&width=960

[61] BLOCKCHAINGAMER.BIZ. ERC-1155 Token Standard [obrázek]. In: Blockchaingamer [online]. 18. června 2018 [cit. 2023-04-07]. Dostupné z: https://www.blockchaingamer.biz/wp-content/uploads/2018/06/erc1155-diagram_750-750x430.jpg

[62] F22 LABS. Screenshot-2022-10-28-at-7.23.32-pm [obrázek]. In: F22 Labs [online]. Bez data [cit. 2023-04-07]. Dostupné z: https://www.f22labs.com/assets/screenshot-2022-10-28-at-7.23.32-pm.png

[63] CRYPTONOMIST. Axie Infinity: A New Paradigm for Blockchain Gaming [obrázek]. In: Cryptonomist [online]. 18. listopadu 2022 [cit. 2023-04-07]. Dostupné z: https://en.cryptonomist.ch/wp-content/uploads/2022/11/axie-infinity-1140x600.jpg.webp

[64] PLAY TO EARN DIARY. party-min [obrázek]. In: Play to Earn Diary [online]. Bez data [cit. 2023-04-07]. Dostupné z: https://playtoearndiary.com/wp-content/uploads/2022/07/party-min.png

[65] GeeksforGeeks. Difference between Web Application and Website [tabulka]. In: GeeksforGeeks [online]. Bez data [cit. 2023-04-07]. Dostupné z: https://www.geeksforgeeks.org/difference-between-web-application-and-website/

[66] PENTEST-TOOLS.COM. Website vulnerability scanning: XSS scanner online [online]. [cit. 2023-05-17]. Dostupné z: https://pentest-tools.com/website-vulnerability-scanning/xss-scanner-online.

[67] AXIE INFINITY. [online]. [cit. 2023-05-17]. Dostupné z: https://axieinfinity.com/.

[68] MAPBOX. [online]. [cit. 2023-05-17]. Dostupné z: https://www.mapbox.com/.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

API	Application Programming Interface
AWS	Amazon Web Services
BaaS	Blockchain as a Service
BAYC	Bored Ape Yacht Club
BNB	Binance Coin
DApp	Decentralized Application
DOM	Document Object Model
ERC	Ethereum Request for Comments
EIP	Ethereum Improvement Proposal
ETH	Ethereum
EVM	Ethereum Virtual Machine
FT	Fungible Token
GDPR	General Data Protection Regulation
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
JSX	JavaScript XML
KYC	Know Your Customer
MVC	Model-View-Controller
NFT	Non-Fungible Token
P2P	Peer-to-Peer
POS	Proof of Stake
POW	Proof of Work
SHA	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol

SQL	Structured Query Language
VIP	Very Important Person
XSS	Cross-Site Scripting

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: OBSAH CD

Obsah CD:

- MovementPackage – Unity package pro pohyb hráče po mapě
- UIPackage – Unity package pro práci s UI
- LandingPageReport – Penetrační report landing page
- MintingPageReport – Penetrační report minting page
- MintingPageWebApp – Webová aplikace, která slouží pro minting tokenů
- LandingPageWebApp – Webová aplikace, poskytující základní informace o projektu
- NFTContract – Solidity kontrakt s deploy scriptem.
- BackendApp – Backendová aplikace