Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)

Кафедра *Системотехніки*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(освітньо-кваліфікаційний рівень)

Дослідження технології блокчейн для проектування та розробки автоматизованих систем у мистецтві
(тема)

Виконав:
студент II курсу, групи *СПРм-22-2*
Суворов М.В..
(прізвище, ініціали)
Спеціальність 122 – «Комп'ютерні науки та інформаційні технології»
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системне проєктування
(повна назва освітньої програми)
Керівник доц. Ситнікова П.Е.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ *Гребеннік І.В.*
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Системотехніки

Рівень вищої освіти другий (магістерський)

Спеціальність 122 «Комп'ютерні науки та інформаційні технології»

Тип програми освітньо-наукова

Освітня програма системне проєктування

ЗАТВЕРДЖУЮ:

Зав. кафедри *СТ*

проф. Гребеннік І.В.

" _____ " *2024 р.*

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Суворову Максиму Віталійовичу*

(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження технології блокчейн для проектування та розробки автоматизованих систем у мистецтві*
затверджена наказом по університету від "*01*" *квітня* 2024р. № *259Ст*

2. Термін подання студентом роботи до екзаменаційної комісії *19 червня 2024 р.*

3. Вихідні дані до роботи *Дослідити та проаналізувати блокчен технологі для проектування та розробки автоматизованих систем у мистецтві. Перелік використовуваних програмних засобів:,. Методи та механізми застосування технології блокчейн у мистецтві, підходи до створення смарт-контрактів. Перелік використовуваних програмних засобів: VSCode, Remix IDE, ОС Microsoft Windows 10. Технічне забезпечення: IBM-сумісний ПК з МП Pentium II або вище.*

4. Перелік питань, що потрібно опрацювати в роботі
4.1 Аналіз проблемної області. 4.1.1 Технологія блокчейн: особливості, історія та розвиток. 4.1.1.1 Поняття технології блокчейн та її особливості. 4.1.1.2 Історія виникнення та розвитку технології блокчейн. 4.1.1.3 Види консенсусу. 4.1.1.4 Типи блокчейнів. 4.1.2. Технологія NFT: використання технології блокчейн у мистецтві. 4.1.2.1 Поняття технології NFT та її особливості. 4.1.2.2 Порівняння стандартів NFT: ERC 721 проти 1155. 4.1.3 Аналіз існуючих методів реалізації технологій блокчейн та NFT у мистецтві. 4.1.3.1 Напрямки застосування блокчейну у мистецтві. 4.1.3.2 Готові рішення для реалізації технології блокчейн у мистецтві. 4.1.4 Постановка завдання. 4.2 Розробка смарт-контракту для вебзастосунку NFT-маркетплейсу. 4.2.1 Огляд технологій розробки. 4.2.1.1 Мова Solidity 4.2.1.2 Мова Yul. 4.2.1.3 Інструмент Hardhat. 4.2.1.4 Remix IDE. 4.2.1.5 Alchemy. 4.2.2 Розробка смарт-контракта NFT-маркетплейса. 4.2.3 Розробка смарт-контракта англійського аукціону. 4.2.4 Розробка смарт-контракта голландського аукціону. 4.2.5 Оптимізація за допомогою Yul. 4.2.6 Розгортання контракту. 4.3 Проектування та розробка вебзастосунку NFT-маркетплейсу. 4.3.1 Огляд фреймворку для розробки. 4.3.1.1 Фреймворк React. 4.3.1.2 Мова каскадних таблиць стилів (CSS). 4.3.1.3 Next.js. 4.3.1.4

Ethers.js. 4.3.1.5 Web3modal. 4.3.1.6 Axios. 4.3.1.7 MetaMask. 4.3.1.8 Pinata. 4.3.2 Функціонал вебзастосунку NFT-маркетплейсу. 4.3.3 Діаграми послідовностей дій (Sequence Diagram) вебзастосунку. 4.3.4 Розробка вебсайту для взаємодії зі смарт-контрактом на блокчейні. 4.3.5 Результати розробки вебсайту для взаємодії зі смарт-контрактом на блокчейні. 4.4 Висновки. 4.5 Перелік посилання.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій)*5.1 Схема блокчейну. 5.2 Середній об'єм торгів та приклад продажів ліквідної колекції . 5.3 Розгортання контракту. 5.4 Закодований рядок. 5.5 Візуальне представлення пам'яті. 5.6 Розгортання отпимізованого контракту. 5.7 Запуск ноди. 5.8 Хxxxxxxxxxxxx (1 аркуш формату А4). 5.9 Розгортання смарт-контракта у мережі Sepolia. 5.10 Діаграма прецедентів NFT-маркетплейсу. 5.11 Sequence Diagram створення NFT . 5.12 Sequence Diagram виставлення NFT на продаж. 5.13 Sequence Diagram купівлі NFT. 5.14 Дерево файлів.*

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів роботи	Термін виконання етапів	Примітка
1.	*Отримання завдання кваліфікаційної роботи*	*01.04.24*	*Виконано*
2.	*Аналіз завдання, літератури та аналогів з теми кваліфікаційної роботи*	*01.04 – 10.04.24*	*Виконано*
3.	*Постановка завдання*	*11.04 – 13.04.24*	*Виконано*
4.	*Розробка смарт-контракта*	*14.04 – 29.04.24*	*Виконано*
3.	*Оптимізація смарт-контракта*	*30.04 – 01.05.24*	*Виконано*
4.	*Проектування вебзастосунку*	*02.05 – 11.05.24*	*Виконано*
5.	*Розробка та тестування програми*	*12.05 – 05.06.24*	*Виконано*
6.	*Оформлення додатків*	*06.06 – 08.06.24*	*Виконано*
7.	*Оформлення пояснювальної записки*	*08.06 – 13.06.24*	*Виконано*
8.	*Оформлення графічної частини та презентаційних матеріалів комп'ютерного захисту*	*14.06 – 16.06.24*	*Виконано*
9.	*Представлення на рецензування*	*14.06.24*	*Виконано*
10.	*Подання роботи до екзаменаційної комісії*	*19.06.24*	*Виконано*

Дата видачі завдання *01 квітня 2024 р.*

Студент _____
(підпис)

Керівник роботи _____ доцент Ситнікова П.Е.
(підпис)

РЕФЕРАТ

Пояснювальна записка до магістерської кваліфікаційної роботи: 85 с., 36 рис., 2 додатки, 34 джерел інформації.

БЛОКЧЕЙН, ДЕЦЕНТРАЛІЗАЦІЯ, ДЕЦЕНТРАЛІЗОВАНА СИСТЕМА, ТРАНЗАКЦІЯ, МЕРЕЖА, РОЗПОДІЛЕНА БАЗА ДАНИХ, СМАРТ-КОНТРАКТИ, ТОКЕН, МИСТЕЦТВО, WEB3, ETHEREUM, EVM, NFT

Об’єктом досліджень є технології блокчейн і NFT та існуючі методи їх реалізації у сфері мистецтва.

Предметом досліджень є інформаційні технології і програмні методи створення клієнтської і серверної частин інформаційної системи, що дозволяє автоматизувати процес створення та тогрівлі предметами мистецтва.

Мета досліджень: аналіз впливу технології блокчейн та невзаємозамінних токенів на розвиток автоматизованих систем у мистецтві, визначення їх можливостей і перспектив у цій галузі.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання бізнес-процесів, методи моделювання розподілених систем, методи оцінки та вибору блокчейн-платформ, методи дослідження проблем безпеки.

У роботі проведено аналіз предметної області, що належить до застосування технологій блокчейн та NFT у сфері мистецтва. Розглянуто особливості та різні типи блокчейнів та механізми консенсусу. Досліджено технологію NFT, її переваги та недоліки, різні стандарти NFT. Проведено аналіз існуючих методів реалізації технологій блокчейн та NFT у мистецтві. Розглянуто існуючі готові рішення для реалізації технології блокчейн у мистецтві та проаналізовано їхні переваги та недоліки. На основі проведеного аналізу сформульовано вимоги до інформаційної системи, що дозволяє автоматизувати процес створення та торгівлі предметами мистецтва з використанням технології блокчейн та NFT. Розроблено смарт-контракти маркетплейсу та двох видів аукціону. Оптимізовано контракт NFT-маркетплейсу. Спроектовано та розроблено вебсайт для взаємодії зі смарт-контрактом на блокчейні.

ABSTRACT

Master’s Thesis: 85 p., 36 pic., 2 appendices, 34 title.

BLOCKCHAIN, DECENTRALIZATION, DECENTRALIZED SYSTEM, TRANSACTION, NETWORK, DISTRIBUTED DATABASE, SMART CONTRACTS, TOKEN, ART, WEB3, ETHEREUM, EVM, NFT

The object of research is blockchain and NFT technologies and existing methods of their implementation in the field of art.

The subject of research is information technology and software methods for creating client and server parts of an information system that automates the process of creating and heating art objects.

Purpose of the research: to analyze the impact of blockchain technology and non-fungible tokens on the development of automated systems in art, to determine their capabilities and prospects in this area.

Research methods: systematic approach, methods of structural analysis and business process modeling, methods of modeling distributed systems, methods of evaluation and selection of blockchain platforms, methods of researching security issues.

The paper analyzes the subject area related to the use of blockchain and NFT technologies in the field of art. The features and different types of blockchains and consensus mechanisms are considered. The NFT technology, its advantages and disadvantages, and various NFT standards are studied. The existing methods of implementing blockchain and NFT technologies in art are analyzed. The existing ready-made solutions for the implementation of blockchain technology in art are considered and their advantages and disadvantages are analyzed. Based on the analysis, the requirements for an information system that allows automating the process of creating and trading art objects using blockchain and NFT technology are formulated. Smart contracts for the marketplace and two types of auction have been developed. The NFT marketplace contract was optimized. A website for interaction with the smart contract on the blockchain was designed and developed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Blockchain (блокчейн) – розподілений та незмінний цифровий реєстр, який використовується для запису транзакцій та відстеження активів;
EVM – Ethereum Virtual Machine, обчислювальне середовище для виконання смарт-контрактів на блокчейні Ethereum;
IPFS – InterPlanetary File System, децентралізована файлово-адресна мережа для зберігання та обміну файлами;
PoW – Proof of Work (доказ роботи), алгоритм консенсусу, який вимагає від учасників мережі виконання складних обчислювальних завдань для підтвердження та обробки транзакцій;
PoS – Proof of Stake (доказ власності), це алгоритм консенсусу, в якому учасники мережі відбираються для створення нових блоків та підтвердження транзакцій на основі кількості криптовалюти, яку вони утримують;
Public key – відкритий ключ, який використовується для шифрування даних та перевірки цифрових підписів;
Private key – Закритий ключ, який використовується для розшифрування даних та створення цифрових підписів;
RPC – Remote Procedure Call, віддалений виклик процедур, метод взаємодії між клієнтом і сервером;
Smart contract – Програма, що зберігається та виконується в блокчейні, яка автоматично виконує умови контракту;
NFT – Non-fungible token (невзаємозамінний токен), унікальний цифровий актив, що представляє право власності на певний об’єкт;
Газ – одиниця виміру вартості транзакцій на Ethereum Virtual Machine.

ВСТУП

На сьогоднішній день технологія блокчейн широко використовується у різних сферах, але в її застосуванні у мистецтві ще існують значні можливості для розвитку. Деякі проєкти вже використовують блокчейн для створення цифрових колекцій та проведення аукціонів, але існують багато інших можливостей, які можуть бути досліджені та впроваджені. Деякі з цих можливостей включають в себе автоматизовану ліцензію для мистецтва, автоматизовані системи управління правами на мистецькі твори, а також розширену реалізацію мистецтва за допомогою віртуальної та розширеної реальності.

Дослідження має актуальність через те, що мистецтво та технологія блокчейн є двома динамічно розвиваючими галузями, які можуть взаємодіяти та доповнювати одна одну. Впровадження автоматизованих систем у мистецтві за допомогою блокчейн може сприяти розвитку цифрового мистецтва, збільшенню прозорості та безпеки транзакцій у мистецькій сфері, а також створенню нових можливостей для художників та колекціонерів. Враховуючи швидке поширення цифрових технологій та збільшення інтересу до цифрових активів, вивчення технології блокчейн у контексті мистецтва має великий потенціал для інновацій та розвитку галузі [1].

Метою дослідження є аналіз впливу технології блокчейн та невзаємозамінних токенів (NFT) на розвиток автоматизованих систем у мистецтві, а також визначення їх можливостей і перспектив у цій галузі.

Завдання дослідження включає:

– вивчення сучасного стану використання технології блокчейн та NFT в сфері мистецтва;

– аналіз існуючих методів реалізації технологій блокчейн та NFT у мистецтві;

– розробка інформаційної системи, що дозволяє автоматизувати процес створення та торгівлі предметами мистецтва з використанням технології блокчейн та NFT;

– визначення переваг і можливих обмежень використання технології блокчейн та NFT у мистецтві;

– возробка рекомендацій щодо впровадження технології блокчейн та NFT в автоматизовані системи у мистецтві.

Об’єктом дослідження є технології блокчейн і NFT та їх вплив на розвиток автоматизованих систем у мистецтві.

Предметом дослідження є інформаційні технології і програмні методи створення клієнтської і серверної частин інформаційної системи, що дозволяє автоматизувати процес створення та торгівлі предметами мистецтва з використанням технології блокчейн та NFT.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання бізнес-процесів, методи моделювання розподілених систем, методи оцінки та вибору блокчейн-платформ, методи дослідження проблем безпеки.

Наукова новизна отриманих результатів полягає у розробці комплексного підходу до аналізу та оцінки потенціалу технологій блокчейн та NFT для автоматизації процесів у сфері мистецтва. Цей підхід включає в себе систематизацію та узагальнення інформації про різні напрямки застосування блокчейну в мистецтві, порівняльний аналіз існуючих NFT-маркетплейсів, а також формулювання чітких вимог до інформаційної системи, що використовує блокчейн та NFT. Запропонований підхід дозволяє отримати цілісне уявлення про можливості та виклики, пов’язані із застосуванням цих технологій у мистецькій сфері, та сформувати основу для подальших досліджень та розробок.

Практичне значення одержаних результатів полягає у тому, що комплексний аналіз технології блокчейн та NFT, проведений у роботі, дає змогу розробникам інформаційних систем, художникам, колекціонерам та іншим учасникам арт-ринку отримати ґрунтовне розуміння можливостей та викликів, пов’язаних із застосуванням цих технологій у сфері мистецтва, що сприятиме створенню ефективних та інноваційних рішень для цифрового мистецтва.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Технологія блокчейн: особливості, історія та розвиток

1.1.1 Поняття технології блокчейн та її особливості

Блокчейн - це розподілена база даних, яка зберігає інформацію у вигляді «блоків», які постійно додаються у ланцюжок. Ця технологія забезпечує безпеку і надійність даних, оскільки кожен блок містить у собі хеш-код попереднього блоку, що робить його дуже складним для зміни без зміни всієї послідовності блоків [2].

Основними особливостями блокчейну є:

– децентралізація: Відсутність централізованої управлінської авторитетності дозволяє кожному вузлу мережі мати копію бази даних;

– незмінність даних: Блокчейн використовує криптографічний захист, що робить неможливим зміну чи видалення блоків;

– прозорість: Усі дані в блокчейні відкриті для перегляду, що дозволяє стежити за кожною транзакцією;

– безпека: Дані в блокчейні зашифровані, а доступ до них можливий лише за допомогою приватного ключа;

– швидкість та ефективність: Блокчейн може автоматизувати процеси, що дозволяє значно зменшити час та витрати на операції.

Важливо розглянути, як саме працює блокчейн. Користувачі мережі відправляють транзакції, які містять дані про переказ активів. Транзакції збираються у блоки для подальшого додавання до ланцюжка. Кожен блок містить певну кількість транзакцій. Для кожного блоку обчислюється хеш-код, який унікально ідентифікує цей блок. Хеш-код формується на основі вмісту блоку і хеш-коду попереднього блоку. Щоб додати новий блок до ланцюжка, мережа вирішує складну криптографічну задачу, яка вимагає великих обчислювальних ресурсів. Цей процес відомий як «доказ роботи» (Proof of Work). Після того, як хеш-код блоку був знайдений, він підтверджується мережею і додається до ланцюжка. Цей блок тепер стає частиною незмінної послідовності блоків (Рисунок 1.1).

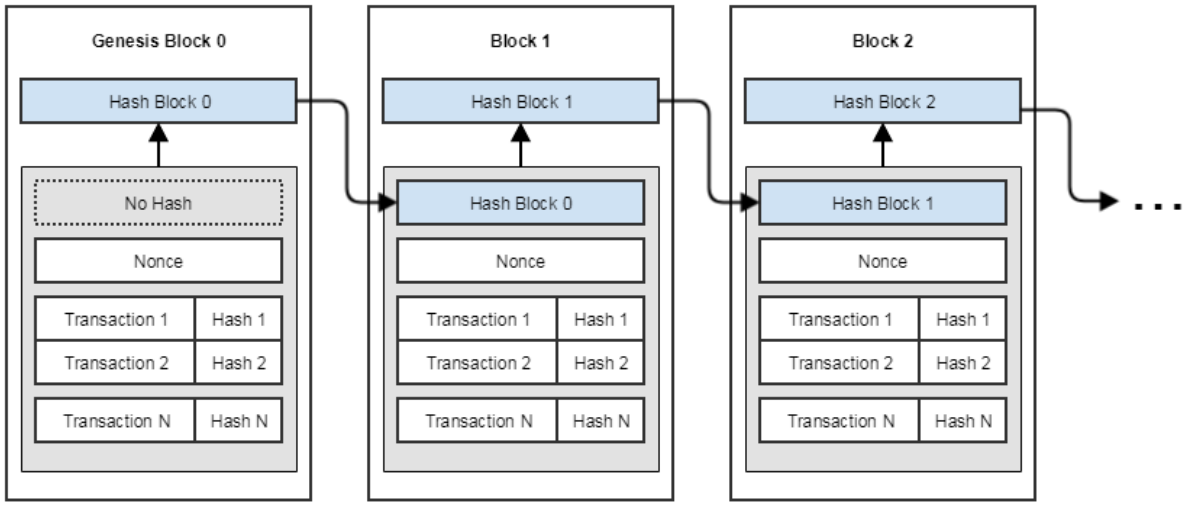

Рисунок 1.1 – Схема блокчейну

Кожен користувач мережі має публічний та приватний ключ. Приватний ключ є секретним числом, яке генерується користувачем і відоме тільки йому. Він використовується для підпису транзакцій і підтвердження власності над активами у мережі блокчейн.

Публічний ключ є відкритим числом, яке створюється на основі приватного ключа за допомогою криптографічних алгоритмів. Цей ключ використовується для підтвердження підписаних повідомлень і для шифрування даних, які можуть бути розшифровані тільки за допомогою відповідного приватного ключа. Важливою особливістю публічного ключа є те, що від нього неможливо вивести вихідний приватний ключ, що гарантує безпеку системи [3].

1.1.2 Історія виникнення та розвитку технології блокчейн

Перші зачатки концепції блокчейну виникли в 1991 році, коли Стюарт Хабер та В. Скотт Стернетт представили концепцію хронологічного ланцюга блоків для захисту даних від впливу сторонніх осіб. Однак ця робота не отримала широкого визнання та залишилася в тіні до майбутніх років.

Справжній прорив у розвитку технології блокчейн стався в 2008 році, коли відомий під псевдонімом Сатоші Накамото опублікував документ, що описував концепцію цифрової криптовалюти Bitcoin та її базову технологію -

блокчейн. Цей документ став витоком для виникнення першої реалізації блокчейну у вигляді Bitcoin.

Після створення Bitcoin і реалізації технології блокчейн у практичному застосуванні, відбулося значне зростання зацікавленості в галузі децентралізованих систем та криптовалют. У 2013 році стартував проєкт Ethereum, який виходив за рамки простого переказу валюти і пропонував концепцію "смарт-контрактів", що стали новим етапом у розвитку блокчейну [4].

З часом технологія блокчейн розширила свої можливості та застосування. Нині блокчейн використовується не лише у фінансовій сфері, але й у сферах логістики, медицини, освіти та мистецтва. Розвиток блокчейну продовжується, і з кожним роком він набуває все більшого значення в сучасному світі.

Після запуску Ethereum у 2015 році виникла можливість створювати смарт-контракти, що є програмним кодом, який автоматично виконує угоди при виконанні певних умов. Це відкрило двері до безлічі нових можливостей використання блокчейну та забезпечило виконання угод без посередництва.

У 2021 році Ethereum запустив оновлення під назвою Ethereum 2.0, що передбачає перехід з механізму консенсусу Proof of Work на Proof of Stake, що має зменшити вплив на навколишнє середовище та покращити масштабованість мережі. Одним із ключових аспектів Ethereum 2.0 є простота впровадження Layer 2 рішень.

Layer 2 (L2) рішення - це механізми, які дозволяють обробляти транзакції поза основним блокчейном Ethereum, забезпечуючи високу швидкість обробки та зниження вартості. Такі рішення включають у себе платформи, які працюють поверх Ethereum і дозволяють виконувати транзакції швидше та дешевше, а потім забезпечують безпеку, переносячи підсумки обробки на основний блокчейн [5].

Ці інновації роблять Ethereum однією з найбільш перспективних блокчейн-платформ для майбутніх розробок у сфері мистецтва, оскільки вони вирішують проблеми масштабованості та вартості, що дозволить створювати більш складні та ефективні додатки та сервіси для цієї галузі.

1.1.3 Види консенсусу

Консенсус – це процес досягнення єдності серед розподілених учасників мережі щодо правильності певної інформації чи стану системи. Консенсусний механізм у блокчейні забезпечує домовленість у визначенні актуального стану мережі та правильності транзакцій. Це досягається шляхом застосування різних алгоритмів, які дозволяють учасникам мережі домовитися про те, який блок (група транзакцій) буде доданий до ланцюжка блоків (блокчейну) наступним.

Proof of Work (PoW) – це алгоритм консенсусу, який використовується в багатьох блокчейн мережах. Основна ідея PoW полягає в тому, щоб доказати, що певна кількість обчислювальних ресурсів була витрачена для створення нового блоку та підтвердження транзакцій у мережі.

Під час роботи алгоритму PoW мережа отримує пакет транзакцій для підтвердження. Майнери у мережі починають проводити обчислення, пробуючи знайти хеш для нового блоку, який відповідає певним умовам. Перший майнер, який успішно знаходить правильний хеш, повідомляє про це всіх учасників мережі. Інші майнери перевіряють правильність хеша та відповідність його умовам. Якщо хеш відповідає вимогам, блок додається до ланцюжка блоків, а майнер отримує винагороду за свою роботу [6]. Схема PoW алгоритму представлена на рисунку А.1.

Переваги PoW:

– захищає мережу від атаки «подвійні витрати». Через високі витрати на обчислювальні ресурси, необхідні для атаки;

– розподіляє владу між учасниками мережі, оскільки кожен може стати майнером.

Недоліки PoW:

– велике споживання електроенергії: обчислення вимагають значних кількостей енергії;

– централізація: майнінг може стати вигідним лише для великих майнінгових ферм та багатих інвесторів. Це може призвести до централізації мережі;

– можливість атаки 51%: якщо одна сторона контролює більше половини обчислювальної потужності мережі, вона може контролювати та маніпулювати мережею.

Proof of Stake (PoS) – це алгоритм консенсусу, який використовується в різних блокчейн мережах для досягнення консенсусу та підтвердження транзакцій. У відміну від Proof of Work, де майнери витрачають обчислювальні ресурси, у PoS вага голосу в мережі залежить від кількості монет, які утримує учасник (Рисунок А.2).

У PoS учасники мережі відправляють свої монети у «стейкінг» (утримання) на певний період часу. Вибирається випадковий учасник, який отримує право створити новий блок на основі його внеску в мережу. Чим більше монет втримує учасник, тим вище його шанс стати обранцем для створення блоку та отримати винагороду [7].

Переваги PoS:

– енергоефективність: PoS вимагає набагато менше енергії порівняно з PoW, оскільки не потребує великих обчислювальних ресурсів;

– зменшення централізації: PoS може бути більш децентралізованим, оскільки не вимагає спеціалізованих майнінгових пристроїв;

– заохочення утримання монет: учасники мережі мають стимул утримувати монети на тривалий термін, щоб отримувати винагороду.

Недоліки PoS:

– можливість атак: атака "відмова в обслуговуванні" (DoS) може бути більш простою в PoS, оскільки атакуючі можуть втратити не таку велику кількість монет;

– ризик централізації: в умовах PoS багаті учасники можуть набути ще більше монет та впливу в мережі.

1.1.4 Типи блокчейнів

Існують кілька типів блокчейнів, які можна класифікувати за різними критеріями, такими як доступність, контроль над консенсусом та рівень децентралізації. Основні типи блокчейнів включають публічні, приватні та консорціальні блокчейни.

Public blockchain (публічні блокчейни) - це блокчейни, до яких має доступ будь-хто, хто бажає приєднатися до мережі, та переглядати або додавати

транзакції. Вони децентралізовані і не потребують довіри до централізованих організацій чи осіб. Прикладом публічного блокчейну є Ethereum.

Private blockchain (приватні блокчейни) - ці блокчейни на відміну від публічних мають обмежений доступ. Лише обрані учасники мережі можуть брати участь у підтвердженні та додаванні нових блоків. Такий тип блокчейну часто використовується у корпоративному середовищі, де необхідно обмежити доступ до даних.

Consortium blockchain (блокчейни консорціуму) - це комбінація публічного та приватного блокчейнів, де кілька організацій або учасників контролюють підтвердження та додавання блоків. Консорціальні блокчейни надають більше контролю над мережею, але все ще зберігають децентралізований характер [8].

Вибір типу блокчейну залежить від конкретних потреб та цілей проєкту. Для глобальних інноваційних проєктів, які прагнуть до максимальної децентралізації та відкритості, підходить публічний блокчейн. Для комерційних застосувань, де важлива конфіденційність та обмежений доступ, приватні або консорціальні блокчейни будуть більш підходящими варіантами.

1.2 Технологія NFT: використання технології блокчейн у мистецтві

1.2.1 Поняття технології NFT та її особливості

NFT (Non-Fungible Token) - це тип криптовалютного токена, який відмінний від звичайних токенів тим, що кожен NFT унікальний та не може бути замінений на інший токен того ж самого типу. Це дає можливість створювати та торгувати унікальними цифровими об’єктами, такими як мистецькі твори, музика, відео, графіка, власні права на нерухомість тощо.

Основними перевагами NFT є:

– унікальність і автентичність: NFT дозволяють ідентифікувати та підтверджувати власність унікальних цифрових активів, що робить їх цінними для колекціонерів та інвесторів;

– легальність і безпека: Блокчейн забезпечує безпеку та непідробність NFT, що важливо для їх легального обігу та власності;

– легкість обігу: NFT можна легко купувати, продавати та обмінювати, використовуючи цифрові ринки та платформи, що сприяє розвитку цифрової економіки;

– власність та контроль: Власник NFT має повний контроль над своїм активом і може вирішувати, як його використовувати чи реалізувати;

– підтримка творців контенту: NFT дозволяють творцям контенту отримувати винагороду за свою роботу без посередників.

Серед недоліків можна виділити недоступність для всіх, оскільки для роботи з NFT потрібно мати доступ до цифрового гаманця та знання про криптовалюту, що може бути складним для багатьох користувачів. Також, я к і з будь-якими іншими інвестиціями, існує ризик масової спекуляції на ринку NFT, що може призвести до нестабільності та ризику втрат [9, 10].

1.2.2 Порівняння стандартів NFT: ERC 721 проти 1155

Стандарти NFT є важливою частиною інфраструктури блокчейн для створення та обігу унікальних цифрових активів. Два з найбільш популярних стандартів NFT на Ethereum - ERC-721 та ERC-1155. Вони відіграють ключову роль у розвитку цифрової економіки та мистецтва.

ERC-721 - стандарт дозволяє реалізувати базовий API (Application Programming Interface) для NFT в рамках смарт-контрактів. Цей стандарт надає базову функціональність для відстеження та передачі NFT. Основна відмінність між ERC-20 та ERC-721 полягає в тому, що ERC-20 токени є взаємозамінними, тобто кожен токен одного типу можна замінити на інший токен того ж типу, і їхня вартість однакова. У ERC-721 кожен токен є унікальним і не може бути замінений на інший токен, навіть того ж типу [11].

ERC-721 токени використовуються для створення цифрових активів, які мають унікальні властивості та можуть бути представлені як унікальні предмети чи об’єкти. Цей стандарт використовується для створення NFTs, які використовуються в цифровому мистецтві, гральній індустрії, зборах і різних інших галузях, де важлива унікальність кожного активу.

ERC-1155 - стандарт описує інтерфейс смарт-контракту, який може представляти будь-яку кількість взаємозамінних і невзаємозамінних типів токенів. Існуючі стандарти, такі як ERC-20, вимагають розгортання окремих

контрактів для кожного типу токена. Ідентифікатор токена стандарту ERC-721 являє собою єдиний невзаємозамінний індекс, і групу цих невзаємозамінних елементів розгортають як єдиний контракт із налаштуваннями для всієї колекції. ERC-1155, навпаки, дає змогу кожному ідентифікатору токена представляти новий тип токена, що налаштовується, який може мати свої власні метадані, ресурси та інші атрибути [12].

ERC-1155 дозволяє ефективніше управляти активами, що мають різні властивості, в тому числі масово видавати та обмінювати їх. Цей стандарт є особливо корисним для ігрової індустрії, де можна створювати токени, які представляють різні предмети гри з різними властивостями.

Обираючи між стандартами ERC-721 та ERC-1155, важливо враховувати конкретні потреби та вимоги проєкту. Якщо потрібно створити унікальні та неповторні токени, ERC-721 може буде кращим вибором. Якщо потрібно створити токени, які можна буде обмінювати та використовувати у різних кількостях, ERC-1155 буде більш підходящим варіантом.

1.3 Аналіз існуючих методів реалізації технологій блокчейн та NFT у мистецтві

1.3.1 Напрямки застосування блокчейну у мистецтві

У сучасному світі, де технологічний прогрес невпинно рухається вперед, галузь мистецтва не залишається осторонь від інноваційних змін. Одним з найбільш перспективних напрямків, що здатний трансформувати традиційні підходи у сфері мистецтва, є блокчейн. Це революційна технологія, заснована на принципах децентралізації, прозорості та безпеки.

Блокчейн, який спочатку асоціювався з криптовалютами, все частіше знаходить своє застосування у різноманітних галузях, включаючи мистецтво. Ця розподілена база даних пропонує унікальні можливості для вирішення низки проблем, з якими стикається арт-індустрія, таких як підробки творів мистецтва, порушення авторських прав, відсутність прозорості на арт-ринку та складнощі з залученням інвестицій.

Застосування блокчейну в мистецтві відкриває нові горизонти для художників, колекціонерів, галерей та інших учасників арт-ринку. Від захисту

авторських прав та боротьби з підробками до токенізації творів мистецтва, децентралізованих арт-платформ, цифрового мистецтва та NFT, а також краудфандингу – ця новітня технологія пропонує низку інноваційних рішень, здатних змінити застарілі парадигми в галузі мистецтва.

Ця тема є надзвичайно актуальною та перспективною для дослідження, оскільки дозволяє проаналізувати потенціал блокчейну для трансформації арт-індустрії, виявити його переваги та можливі виклики, а також розглянути конкретні приклади успішного застосування цієї технології в різних напрямках мистецтва.

Розглянемо основні напрямки застосування цієї технології в мистецькій сфері:

а) захист авторських прав та боротьба з підробками. Однією з найбільших проблем у мистецькій індустрії є підробки творів мистецтва та порушення авторських прав. Блокчейн дозволяє створити незмінний цифровий сертифікат автентичності для творів мистецтва, що значно ускладнює можливість підробки. Кожен твір мистецтва може бути зареєстрований у блокчейні, створюючи унікальний цифровий відбиток, який підтверджує авторство та походження твору. Це дозволяє захистити права художників, а також забезпечити прозорість арт-ринку.

б) токенізація творів мистецтва. Блокчейн дозволяє токенізувати твори мистецтва, перетворюючи їх на унікальні цифрові активи. Кожен токен представляє частку власності на твір мистецтва, що відкриває нові можливості для інвестування та торгівлі у сфері мистецтва. Це робить арт-ринок більш доступним для широкого кола інвесторів, оскільки вони можуть придбати частку твору мистецтва, а не повністю володіти ним.

в) децентралізовані арт-платформи. Блокчейн-технологія сприяє створенню децентралізованих арт-платформ, де художники можуть безпосередньо взаємодіяти з колекціонерами та продавати свої твори без посередників. Такі платформи забезпечують прозорість транзакцій, безпечні платежі та захист авторських прав. Це дозволяє художникам зберігати більшу частину доходів від продажу своїх творів і встановлювати безпосередній зв’язок з аудиторією.

г) цифрове мистецтво та NFT. Блокчейн став основою для розвитку цифрового мистецтва та невзаємозамінних токенів. NFT є унікальними

цифровими активами, які представляють право власності на цифрові твори мистецтва, такі як зображення, відео, анімації тощо. Ця технологія дозволяє художникам створювати, продавати та підтверджувати автентичність своїх цифрових творів, а колекціонерам – безпечно їх купувати та володіти [13].

д) краудфандинг. Блокчейн відкриває нові можливості для краудфандингу у сфері мистецтва. Художники можуть випускати токени або криптовалюту, пов'язану з їхніми творами, і продавати їх прихильникам та колекціонерам. Це дозволяє залучати фінансування для нових проєктів, а інвестори можуть отримувати частину доходів або унікальні винагороди від художників.

Застосування блокчейну в мистецтві має потенціал для трансформації традиційної моделі арт-ринку, забезпечуючи більшу прозорість, безпеку та доступність для всіх учасників. Однак, слід зазначити, що ця технологія все ще знаходиться на ранній стадії розвитку, і її повний вплив на мистецьку індустрію ще належить оцінити.

Невзаємозамінні токени часто мають додаткову функціональність, яка називається «ютилітою» (utility). Ютиліта NFT – це додатковий набір переваг або можливостей, що надаються власникам цифрового активу. Розглянемо основні з них:

а) доступ до ексклюзивного контенту або спільноти. Володіння певним NFT може забезпечити власникам доступ до закритих чатів, форумів, подій чи інших ексклюзивних привілеїв, пов'язаних з проєктом або творцем NFT. Наприклад, NFT колекція «Bored Ape Yacht Club» або «BAYC» від Yuga Labs забезпечує власникам доступ до закритого онлайн-форуму та чату лише для власників цих унікальних цифрових персонажів. Також «BAYC» часто влаштовують заходи на яхтах, завітати на які можуть лише власники їх NFT колекції.

б) участь у прийнятті рішень. Деякі NFT проєкти надають право голосу власникам токенів щодо майбутніх оновлень, напрямків розвитку проєкту та інших ключових рішень. Наприклад, власники NFT проєкту «Decentraland» мають право голосу щодо нових оновлень та змін у цьому віртуальному світі, базованому на блокчейні.

в) отримання пасивного доходу. Проєкти можуть передбачати розподіл частини доходів або винагород серед власників NFT, створюючи потенційне джерело пасивного доходу. Наприклад, згідно з «білим папером» проєкту NFT

«Etheria», частина доходів від продажу землі у віртуальному світі буде регулярно розподілятися між власниками відповідних NFT.

г) знижки та привілеї. Власники NFT можуть отримувати знижки на продукти, послуги, наступні випуски колекції NFT від того ж творця або проєкту.

д) гейміфікація та віртуальні світи. У сфері гейміфікації та віртуальних світів NFT можуть надавати доступ до унікальних ігрових активів, аватарів, земельних ділянок або інших ексклюзивних елементів. Наприклад, у блокчейн-грі «Axie Infinity» унікальні NFT представляють персонажів, які використовуються гравцями для боротьби та заробітку токенів [14].

е) комерційні права. В деяких випадках NFT може надавати комерційні права на використання цифрового активу, такі як права на ліцензування або створення похідних товарів чи творів мистецтва. Наприклад, при купівлі NFT серії «EtherRocks» покупець отримує комерційні права на використання зображення цієї породи в комерційних цілях, включаючи створення похідних творів.

ж) паспортизація та підтвердження досягнень. Одним із цікавих напрямків застосування ютиліти NFT є паспортизація та підтвердження досягнень, участі у певних подіях, отримання нагород або дипломів. NFT можуть виступати своєрідними «цифровими паспортами» чи сертифікатами, що містять записи на блокчейні. Наприклад, організатори конференцій чи фестивалів можуть видавати спеціальні NFT усім зареєстрованим учасникам як підтвердження їхньої присутності на заході. Ці унікальні токени будуть містити інформацію про подію, дату, місце проведення тощо і можеть мати вищенаведені ютиліти.

Наявність ютиліти робить NFT більш привабливими для колекціонерів та інвесторів, оскільки вони отримують не лише цифровий рідкісний твір, а й додаткові переваги та можливості, пов’язані з володінням цим активом. Ютиліта NFT є важливим фактором, який може сприяти зростанню цінності та популярності цифрового мистецтва на блокчейні [15].

Розглянемо недоліки існуючих методів реалізації технологій блокчейн та NFT у мистецтві:

– високі комісії за транзакції: Популярні блокчейн-платформи першого рівня «L1», такі як Ethereum, можуть мати високі комісії за транзакції, особливо

в періоди високого навантаження на мережу. Це може стати перешкодою для художників та колекціонерів з обмеженими ресурсами;

– складність використання: Технології блокчейн та NFT можуть бути досить складними для розуміння та використання, особливо для тих, хто не має технічного досвіду. Процеси створення, купівлі та продажу NFT вимагають певних знань та навичок, таких як робота з криптогаманцями, обмінниками криптовалют та взаємодія зі смарт-контрактами. Це може відлякувати деяких художників та колекціонерів, які не готові вивчати нове;

– ризик спекуляцій та цінових бульбашок: У світі NFT та цифрового мистецтва існує ризик утворення спекулятивних бульбашок, коли ціни на певні твори чи колекції надмірно завищуються через спекулятивний попит, а не через реальну художню цінність. Це може призвести до нестабільності ринку та втрати довіри до NFT як цінних активів;

– проблеми масштабованості: Існуючі блокчейн-мережі мають обмежену пропускну здатність, що може стати проблемою при масштабуванні застосування NFT у мистецтві. Коли на мережу надходить велика кількість транзакцій, це призводить до затримок та високих комісій, що може зашкодити ефективності використання NFT;

– правова невизначеність: Правовий статус NFT та питання авторських прав у цифровому середовищі все ще залишаються певною мірою невизначеними в багатьох юрисдикціях. Це створює ризики та невизначеність для художників та колекціонерів, оскільки немає чітких правових рамок для регулювання цієї нової сфери;

– питання вартості цифрового мистецтва: Існують дискусії щодо справжньої цінності цифрових артефактів, що не мають фізичного втілення. Деякі критики ставлять під сумнів обґрунтованість високих цін на NFT та цифрове мистецтво, вважаючи, що вони не мають реальної матеріальної цінності;

– екологічний вплив окремих блокчейнів: Блокчейни, які все ще використовують механізм консенсусу PoW, такі як Bitcoin, споживають величезну кількість енергії для майнінгу, що має значний негативний вплив на навколишнє середовище. Однак провідні платформи вже перейшли на більш екологічний PoS.

Незважаючи на ці недоліки, технології блокчейн та NFT продовжують розвиватися, а нові рішення та підходи можуть допомогти подолати ці виклики. Однак важливо ретельно розглядати ці недоліки та шукати способи їх усунення для забезпечення стійкого та відповідального застосування цих технологій у галузі мистецтва [16].

1.3.2 Готові рішення для реалізації технології блокчейн у мистецтві

З появою технології блокчейн та NFT з’явилась нова можливість для митців та колекціонерів цифрового мистецтва створювати, продавати та купувати унікальні твори у цифровому середовищі. NFT маркетплейси стали своєрідними майданчиками, що об’єднують творців, шанувальників мистецтва та інвесторів навколо торгівлі цифровими активами, верифікованими на блокчейні.

Ці онлайн-платформи дозволяють митцям перетворювати свої роботи на унікальні цифрові активи, захищені технологією блокчейн від підробок та множинних копій. Колекціонери можуть придбати та продемонструвати своє цифрове мистецтво, одночасно підтримуючи митців роялті від подальших перепродажів. А інвестори можуть спекулювати на цінності рідкісних NFT, як це відбувається на традиційному арт-ринку.

Хоча створення власного NFT маркетплейсу залишається складним і дорогим завданням, з’явилися готові рішення, що спрощують процес впровадження технології блокчейн у світ цифрового мистецтва. Ці платформи пропонують різноманітні можливості та переваги, на які митці та колекціонери мають звертати увагу при виборі відповідного рішення.

Поява NFT-маркетплейсів стала відповіддю на зростаючий попит на зручні та безпечні платформи для торгівлі цифровим мистецтвом. Піонерами у цій сфері стали такі проекти, як OpenSea та Rarible, запущені у 2017-2018 роках. Вони запропонували художникам можливість перетворювати свої цифрові твори на унікальні токени на блокчейні Ethereum та продавати їх колекціонерам з усього світу.

З часом NFT-маркетплейси еволюціонували, пропонуючи більш досконалі функції та спеціалізовані рішення для різних сегментів арт-ринку. Сьогодні ці платформи є готовими рішеннями для реалізації технології

блокчейн у мистецтві, надаючи зручний інтерфейс, широкий вибір творів та спільноти, присвяченій цифровому мистецтву. Наразі, середній об’єм тижневих продажів дорівнює приблизно 1000 ETH, що на момент написання є еквівалентом 4 млн доларів (Рисунок 1.2).

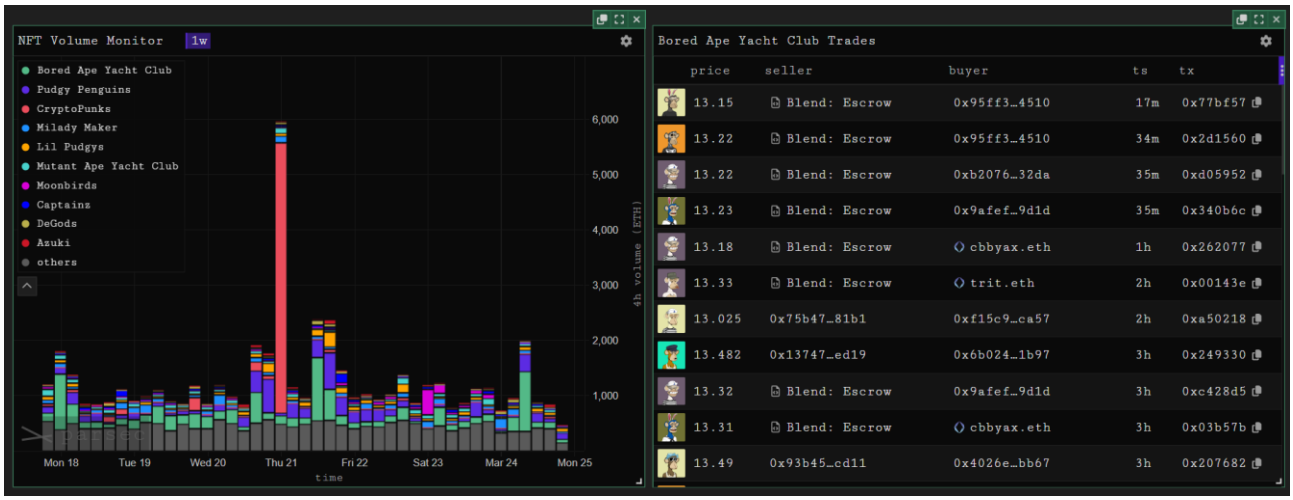

Рисунок 1.2 – Середній об’єм торгів та приклад продажів ліквідної колекції

Розглянемо детальніше деякі популярні готові рішення для реалізації технології блокчейн у мистецтві, їхні ключові переваги та недоліки.

«BinanceNFT» запущений однією з найбільших криптобірж у світі - Binance. Цей NFT-маркетплейс пропонує широкий вибір творів мистецтва, колекційних предметів та ігрових активів, представлених у вигляді NFT. «BinanceNFT» інтегрований з екосистемою Binance, що спрощує покупку та продаж NFT за допомогою криптовалют. Платформа також має зручний інтерфейс для перегляду та пошуку NFT, а також можливості для створення власних колекцій (Рисунок 1.3).

Переваги:

– висока ліквідність завдяки великій користувацькій базі Binance;

– зручний інтерфейс для створення, купівлі та продажу NFT;

– безпечність завдяки репутації Binance.

Недоліки:

– централізована платформа, що суперечить ідеї децентралізації блокчейну;

– обмежені можливості для кастомізації та розширення функціоналу;

– підтримує лише три блокчейни;
– орієнтований більше на масовий ринок, ніж на високе мистецтво.

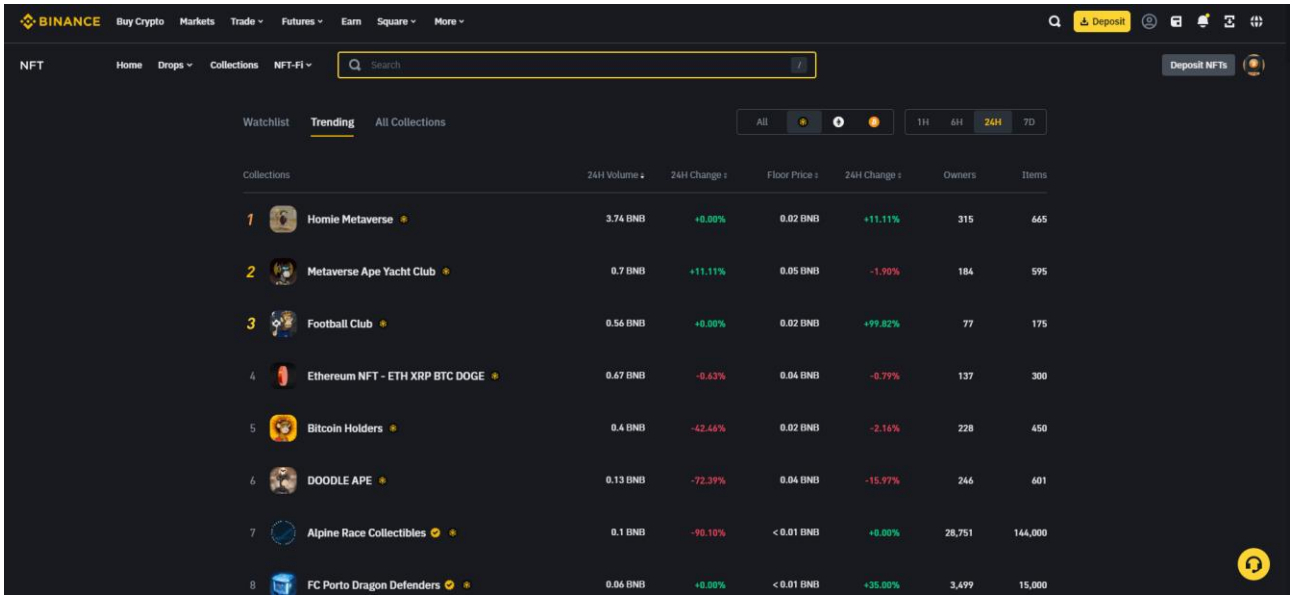

Рисунок 1.3 – Інтерфейс «BinanceNFT»

«Zora.co» – децентралізований маркетплейс для NFT на основі Ethereum, який фокусується на творчості та мистецтві. Ця платформа пропонує безкоштовний і відкритий код, що дозволяє митцям і розробникам створювати власні продукти та додатки на її основі (Рисунок 1.4).

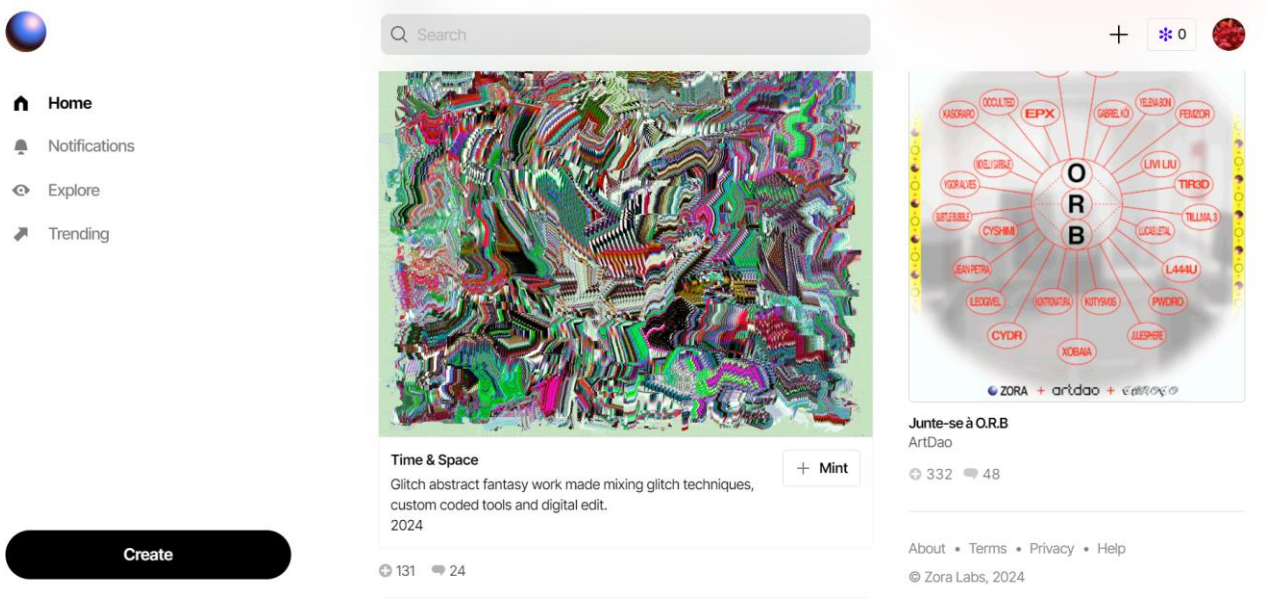

Рисунок 1.4 – Інтерфейс «Zora.co»

Переваги:

– децентралізований, що забезпечує більшу прозорість;

– зосередженість на високому мистецтві та незалежних митцях;

– має роялті для митців.

Недоліки:

– складніший інтерфейс для непідготовлених користувачів;

– обмежені можливості інтеграції з іншими блокчейнами;

– обов’язкова комісія платформи 0.000777 ETH на кожну покупку;

– відсутність можливості перепродажу придбаних токенів;

– відсутня можливість проведення аукціонів.

«TofuNFT» – це NFT-маркетплейс, сфокусований на азійському ринку цифрового мистецтва. Платформа пропонує широкий вибір творів від художників з Азії, а також має локалізовані інтерфейси для різних країн регіону. «TofuNFT» також має функції для створення та продажу власних NFT, а також організації аукціонів та випуску обмежених колекцій (Рисунок 1.5).

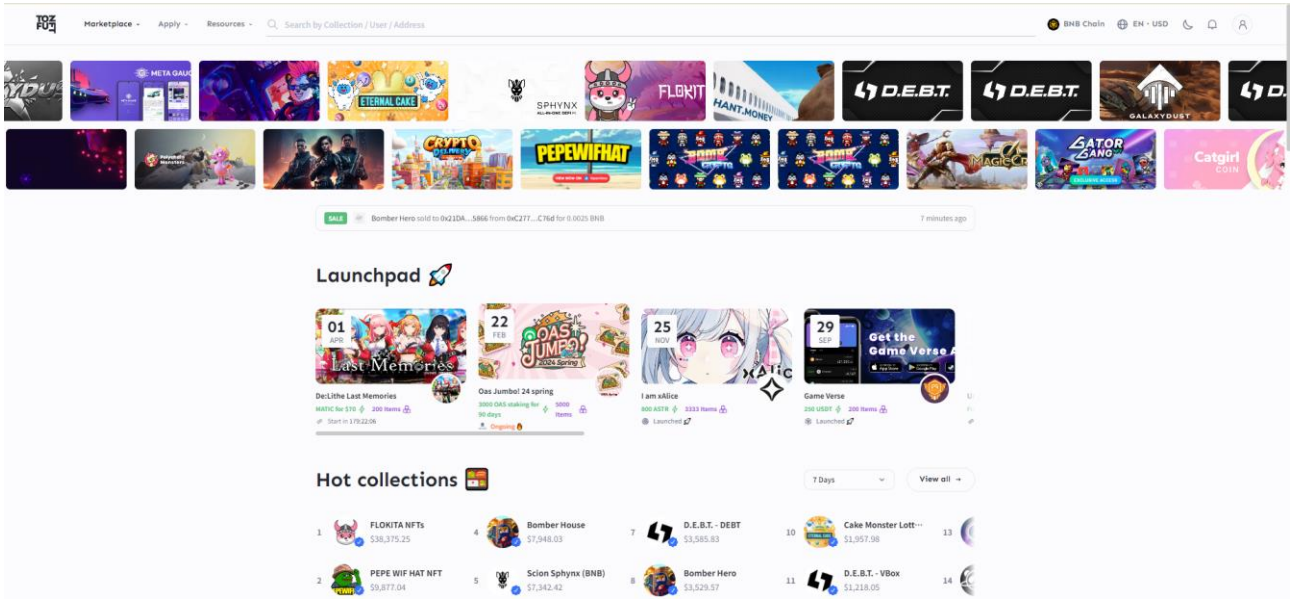

Рисунок 1.5 – Інтерфейс «TofuNFT»

Переваги:

– тісна співпраця з провідними криптохудожниками;

– акцент на творчості та наративі навколо творів;

– підтримка різних стандартів NFT (ERC-721, ERC-1155 тощо);

– має двадцять одну мережу для торгівлі.

Недоліки:

– обмежені можливості для маркетингу та просування NFT;

– відсутня можливість створення власних токенів та їх подальшого продажу;

– торгувати можна лише картинками.

Готові рішення NFT-маркетплейсів відіграють важливу роль у впровадженні технології блокчейн у галузі мистецтва. Вони надають зручну інфраструктуру для художників, колекціонерів та любителів мистецтва, забезпечуючи безпечний і прозорий спосіб створювати, купувати, продавати та торгувати цифровими творами [17].

Однак, незважаючи на численні переваги цих платформ, існують також певні недоліки, які необхідно вирішувати. Серед них – масштабованість, комісійні збори, проблеми з інтерфейсом та доступністю для непідготовлених користувачів. Більшість наявних рішень все ще орієнтовані на конкретні ніші або цільові аудиторії. Потрібні більш універсальні платформи, здатні охопити широкий спектр творчих напрямків та забезпечити рівний доступ для всіх бажаючих.

1.4 Постановка завдання

Враховуючи аналіз існуючих методів реалізації технологій блокчейн та NFT у мистецтві, а також їхні недоліки, завданням даного проєкту є розробка інформаційної системи, що дозволяє автоматизувати процес створення та торгівлі предметами мистецтва з використанням технології блокчейн та NFT, усуваючи при цьому недоліки існуючих рішень.

Система повинна відповідати наступним вимогам:

– система має бути побудована на децентралізованій блокчейн-платформі, що забезпечить прозорість, безпеку та незалежність від централізованих органів управління;

– система повинна підтримувати роботу з різними блокчейн-мережами, що розширить можливості для художників та колекціонерів, а також зменшить залежність від однієї платформи;

– система має бути універсальною та охоплювати широкий спектр творчих напрямків, включаючи живопис, скульптуру, музику, відео, фотографію, цифрові твори та інші форми мистецтва;

– система повинна надавати можливість проводити аукціони з продажу NFT, використовуючи різні формати, такі як англійський та голландський аукціони. Це дозволить підвищити ефективність торгівлі та залучити більше учасників;

– система повинна використовувати методи оптимізації смарт-контрактів, такі як Yul, для зменшення комісій за транзакції та підвищення доступності платформи для художників та колекціонерів;

– система повинна мати інтуїтивно зрозумілий та зручний інтерфейс, який буде доступний для користувачів з різним рівнем технічних знань. Це забезпечить широке охоплення аудиторії та спростить взаємодію з платформою;

Розробка такої інформаційної системи сприятиме розвитку цифрового мистецтва, підвищенню прозорості та безпеки транзакцій у мистецькій сфері, а також створенню нових можливостей для художників та колекціонерів.

2 РОЗРОБКА СМАРТ-КОНТРАКТУ ДЛЯ ВЕБЗАСТОСУНКУ NFT-МАРКЕТПЛЕЙСУ

2.1 Огляд технологій розробки

При створенні смарт-контракту для веб-додатка NFT-маркетплейсу було вирішено використовувати наступні технології:

– Solidity – мова програмування смарт-контрактів;

– Yul – мова програмування для оптимізації смарт-контрактів;

– Hardhat – інструмент для тестування та розгортання смарт-контрактів;

– Remix IDE – інструмент для розробки та розгортання смарт-контрактів;

– Alchemy – платформа для доступу до інфраструктури блокчейна та виконання транзакцій

2.1.1 Мова Solidity

Solidity – це високорівнева мова програмування, яка була створена для написання смарт-контрактів на платформі Ethereum. Смарт-контракти це програми, які автоматизують виконання угод між сторонами без посередництва. Solidity була створена, щоб забезпечити безпеку та ефективність при виконанні цих контрактів [18, 19].

Основними особливостями Solidity є:

– простота: Solidity подібна до JavaScript та розроблена так, щоб бути зрозумілою для розробників, які мають досвід у мовах програмування загального призначення. Вона містить у собі концепції такі, як змінні, функції, цикли та умови, що робить її доступною для розробників з різним рівнем кваліфікації;

– ефективність: Solidity має вбудовані механізми, що дозволяють оптимізувати роботу смарт-контрактів для зменшення витрат газу (одиниця виміру вартості транзакцій на Ethereum) і підвищення швидкодії;

– безпека: Solidity має вбудовані механізми безпеки, що допомагають уникнути деяких типових проблем, які можуть виникнути при розробці смарт-

контрактів, таких як переповнення буфера або втрату грошей через неправильну логіку;

– гнучкість: Solidity підтримує різноманітні шаблони проектування та може бути використана для створення різних типів смарт-контрактів, від фінансових до голосування та багатьох інших;

– сумісність з різними блокчейнами: Solidity інтегрується безпосередньо з Ethereum Virtual Machine (EVM), що дозволяє виконувати смарт-контракти на будь-якому EVM-сумісному блокчейні;

– активна спільнота: Є велика спільнота розробників, яка підтримує Solidity і надає важливу документацію, уроки та підтримку як для новачків, так і для досвідчених розробників.

2.1.2 Мова Yul

Yul – це мова низькорівневого представлення для EVM. Вона була розроблена для написання смарт-контрактів на більш низькому рівні, ніж Solidity що дозволяє оптимізувати витрати газу [20].

Основними особливостями мови Yul є:

– низькорівневість: Yul є ближче до машинного коду та дозволяє розробникам працювати з EVM на більш низькому рівні, ніж може забезпечити Solidity. Це дозволяє більш точно контролювати операції, які виконуються смарт-контрактом;

– ефективність: Мова дозволяє писати оптимізований код, який може бути більш ефективним у виконанні на EVM, що може бути важливо для зменшення витрат газу;

– інтеграція з Solidity: Хоча Yul може використовуватися для написання цілих смарт-контрактів, він часто використовується як доповнення до Solidity, де розробники можуть використовувати Yul для оптимізації певних частин коду.

2.1.3 Інструмент Hardhat

Hardhat – це розширене середовище розробки для Ethereum, яке дозволяє розробникам створювати, тестувати та розгортати смарт-контракти [21].

Основні особливості Hardhat включають:

– підтримка мови Solidity: Hardhat надає підтримку для розробки смарт-контрактів на мові Solidity, що є стандартом для розробки на платформі Ethereum;

– локальний вузол Ethereum: Hardhat включає в себе локальний вузол Ethereum, що дозволяє розробникам тестувати свої смарт-контракти безпосередньо на своєму комп'ютері;

– вбудований тестувальний фреймворк: Hardhat має вбудований тестувальний фреймворк, який дозволяє розробникам писати та запускати автоматизовані тести для своїх смарт-контрактів;

– підтримка мереж Ethereum: Hardhat підтримує різні мережі Ethereum, включаючи тестові мережі, основну мережу та всі інші EVM-сумісні мережі;

– розширені можливості розгортання: Hardhat надає розширені можливості для розгортання смарт-контрактів, включаючи підтримку різних протоколів розгортання та інтеграцію з іншими сервісами;

– підтримка плагінів: Hardhat дозволяє розширювати свої можливості за допомогою плагінів, що дозволяє розробникам налаштовувати середовище роботи під свої потреби.

2.1.4 Remix IDE

Remix IDE – це інтегроване середовище розробки для смарт-контрактів на блокчейні Ethereum. Воно надає розробникам зручний спосіб створювати, відлагоджувати та тестувати смарт-контракти безпосередньо у веб-браузері [22].

Основними особливостями Remix IDE є:

– вбудована підтримка Solidity та Yul: Remix IDE має вбудовану підтримку мови Solidity та Yul, що дозволяє розробникам легко створювати смарт-контракти без необхідності встановлення додаткових інструментів;

– інтерактивна консоль: У Remix IDE є інтерактивна консоль, яка дозволяє розробникам взаємодіяти з їх смарт-контрактами безпосередньо з інтерфейсу IDE;

– візуальне представлення контрактів: Remix IDE надає можливість візуального представлення структури та взаємодії між смарт-контрактами, що полегшує розуміння та відлагодження коду;

– інтеграція з різними середовищами виконання: Remix IDE підтримує інтеграцію з різними середовищами виконання, такими як локальний вузол Ethereum, тестова мережа, основна мережа Ethereum та всі інші EVM-сумісні мережі;

– підтримка плагінів: Remix IDE дозволяє розширювати можливості за допомогою плагінів, що дозволяє розробникам налаштовувати середовище роботи під свої потреби.

2.1.5 Alchemy

Alchemy – це платформа для розробників блокчейну, яка надає інструменти та сервіси для роботи з різними EVM-сумісними блокчейнами.

Основні особливості Alchemy включають:

– хмарна інфраструктура для блокчейну: Alchemy надає хмарну інфраструктуру для блокчейну, що дозволяє розробникам легко створювати, масштабувати та управляти додатками на блокчейні;

– інтерфейс для взаємодії з блокчейном: Alchemy надає інтерфейс для взаємодії з блокчейном, що дозволяє розробникам створювати та виконувати транзакції, взаємодіяти зі смарт-контрактами та отримувати дані з блокчейну;

– інструменти розробки: Alchemy надає різноманітні інструменти розробки, такі як інтерфейси API, SDK та бібліотеки, що дозволяють розробникам швидко створювати та розгортати додатки на блокчейні;

– моніторинг та аналітика: Alchemy надає інструменти для моніторингу та аналітики додатків на блокчейні, що дозволяє розробникам відстежувати та аналізувати роботу своїх додатків.

2.2 Розробка смарт-контракта NFT-маркетплейса

Смарт-контракт – це самовиконуваний контракт, де умови угоди між сторонами написані безпосередньо в рядках коду. Цей код і угоди, які він містить, зберігаються на децентралізованій блокчейн-мережі. Смарт-контракти автоматизують виконання угод, забезпечуючи надійність та прозорість, оскільки вони виконуються безпосередньо блокчейн-мережею і не потребують посередників. При настанні певних умов, прописаних у коді смарт-контракту, відбувається автоматичне виконання відповідних дій, таких як переказ токенів або передача прав власності.

Смарт-контракти забезпечують високу ступінь безпеки та незмінності, оскільки після їх розгортання в блокчейні, зміна коду контракту або його умов стає надзвичайно складною. Це гарантує, що всі сторони можуть бути впевнені в виконанні угоди відповідно до попередньо визначених умов. Використання смарт-контрактів також усуває ризик людських помилок і значно прискорює процес виконання угод. Вони можуть застосовуватися у різних галузях, включаючи фінанси, страхування, логістику, управління ланцюгами поставок, нерухомість та багато інших сфер, де необхідна автоматизація і надійність договірних відносин.

Перед початком написання контракту вказується ліцензія. Оскільки в нашому випадку код відкритий, то вказується ліцензія MIT. Ліцензія вказується на першому рядку для забезпечення відповідності вимогам відкритого ліцензування.

Далі імпортуються необхідні бібліотеки та контракти з OpenZeppelin, які надають стандартизовані функціональності для роботи з лічильниками, токенами ERC721, та розширенням ERC721URIStorage, що дозволяє зберігати URI (метадані) для кожного токена (Лістинг 2.1).

Лістинг 2.1 – Ліцензування та імпортування бібліотек

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;
import "@openzeppelin/contracts/utils/Counters.sol";
import
"@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
```

Далі вказується ключове слово contract, назва контракту NFTMarketplace, контракт від якого успадковується (ERC721URIStorage) та фігурні дужки, у яких пишеться вся логіка контракту. Успадкування ERC721URIStorage здійснюється для роботи з NFT токенами, які мають метадані. Також одразу використовуємо бібліотеку Counters для лічильників токенів та проданих предметів. Визначаємо змінну listingPrice, яка вказує вартість виставлення предмета на маркетплейсі, та змінну owner, яка зберігає адресу власника маркетплейсу (Лістинг 2.2).

Лістинг 2.2 – Створення контракту, лічильників, вартість виставлення на продаж та адресу власника

```
contract NFTMarketplace is ERC721URIStorage {
using Counters for Counters.Counter;
Counters.Counter private _tokenIds;
Counters.Counter private _itmesSold;
uint256 listingPrice = 0.00015 ether;
address payable owner;
}
```

У лістингу 2.3 наведено визначення мапінгу для зберігання ринкових предметів за їх ідентифікаторами та структура MarketItem, яка описує ринковий предмет з його атрибутами (ідентифікатор токена, адреса продавця, адреса власника, ціна токена та статус продажу токена). Також наведено оголошення події idMarketItemCreated, яка буде викликатися при створенні ринкового предмета.

Лістинг 2.3 – Структура та івент

```
mapping(uint256 => MarketItem) private idMarketItem;
struct MarketItem {
    uint256 tokenId;
    address payable seller;
    address payable owner;
    uint256 price;
    bool sold;
}
event idMarketItemCreated(
```

```
        uint256 indexed tokenId,
        address seller,
        address owner,
        uint256 price,
        bool sold
    );
```

У лістингу 2.4 вказано модифікатор onlyOwner, який обмежує доступ до функцій лише власнику маркетплейсу. Конструктор контракту, який встановлює власника маркетплейсу та викликає конструктор батьківського контракту ERC721. Функція updateListingPrice, яка дозволяє власнику маркетплейсу змінювати вартість виставлення предмета на продаж та функція getListingPrice, яка повертає поточну вартість виставлення предмета на продаж.

У Solidity, «_» є спеціальним символом, який використовується в контексті модифікаторів функцій для позначення місця, де основний код функції буде виконаний. Тобто функція updateListingPrice виконається лише після перевірки модифікатора.

Лістинг 2.4 – Змінення та повернення вартості NFT

```
modifier onlyOwner() {
      require(
          msg.sender == owner,
          "only owner of the marketplace can change the listing price"
      );
      _;
  }
  constructor() ERC721("NFT Token", "MYNFT") {
      owner == payable(msg.sender);
  }
  function updateListingPrice(uint256 _listingPrice)
      public
      payable
      onlyOwner
  {
      listingPrice = _listingPrice;
  }
  function getListingPrice() public view returns (uint256) {
      return listingPrice;    }
```

Приватна функція createMarketItem, яка створює ринковий предмет, додає його до мапінгу, а також передає токен контракту (Лістинг 2.5). Також присутня перевірка, що ціна більше нуля та перевірка, що ціна дорівнює вартості виставлення, які спрацьовують під час підписання транзакції.

Лістинг 2.5 – Функція створення ринкового предмету

```
function createMarketItem(uint256 tokenId, uint256 price) private {
   require(price > 0, "Price must be at least 1");
   require(msg.value == listingPrice,
       "Price must be equal to listing price");
   idMarketItem[tokenId] = MarketItem(
       tokenId,
       payable(msg.sender),
       payable(address(this)),
       price,
       false
   );
   _transfer(msg.sender, address(this), tokenId);
   emit idMarketItemCreated(
       tokenId,
       msg.sender,
       address(this),
       price,
       false);}
```

Функцію createToken, яка створює новий токен (NFT) та викликає функцію створення ринкового предмета для цього токена представлено у лістингу 2.6.

Лістинг 2.6 – Функція створення NFT

```
function createToken(string memory tokenURI, uint256 price)
      public
      payable
      returns (uint256)
  {
      _tokenIds.increment();
      uint256 newTokenId = _tokenIds.current();
```

```
        _mint(msg.sender, newTokenId);
        _setTokenURI(newTokenId, tokenURI);
        createMarketItem(newTokenId, price);
        return newTokenId;
    }
```

Функція reSellToken дозволяє власнику токена виставити його на повторний продаж. Вона перевіряє, чи власник токена є тим, хто викликає функцію, і чи вартість виставлення на продаж дорівнює встановленій ціні листингу. Якщо ці перевірки проходять успішно, функція змінює статус токена на "незареєстрований" і оновлює інформацію про ціну і власника (Лістинг 2.7).

Лістинг 2.7 – Функція для перепродажу токена

```
function reSellToken(uint256 tokenId, uint256 price) public payable {
        require(
            idMarketItem[tokenId].owner == msg.sender,
            "Only item owner can perform this operation"
        );
        require(
            msg.value == listingPrice,
            "Price must be equal to listing price"
        );
        idMarketItem[tokenId].sold = false;
        idMarketItem[tokenId].price = price;
        idMarketItem[tokenId].seller = payable(msg.sender);
        idMarketItem[tokenId].owner = payable(address(this));
        _itmesSold.decrement();
        _transfer(msg.sender, address(this), tokenId);
    }
```

Функція createMarketSale дозволяє покупцеві придбати токен, виставлений на продаж. Вона перевіряє, чи передана вартість дорівнює ціні токена, після чого оновлює інформацію про власника і статус токена, здійснює транзакції і передає токен покупцеві (Лістинг 2.8).

Лістинг 2.8 – Функція придбання токена

```
function createMarketSale(uint256 tokenId) public payable {
        uint256 price = idMarketItem[tokenId].price;
```

```
        require(
            msg.value == price,
            "Please submit the asking price in order to complete the
purchase"
        );
        idMarketItem[tokenId].owner = payable(msg.sender);
        idMarketItem[tokenId].sold = true;
        idMarketItem[tokenId].owner = payable(address(0));
        _itmesSold.increment();
        _transfer(address(this), msg.sender, tokenId);
        payable(owner).transfer(listingPrice);
        payable(idMarketItem[tokenId].seller).transfer(msg.value);
    }
```

Функція fetchMarketItem повертає масив токенів, які ще не були продані і все ще знаходяться на ринку. Вона проходить через всі створені токени і вибирає ті, що мають власника address(this), тобто адресу смарт-контракта (Лістинг 2.9).

Лістинг 2.9 – Функція, яка повертає не продані токени

```
  function fetchMarketItem() public view returns (MarketItem[] memory) {
        uint256 itemCount = _tokenIds.current();
        uint256 unSoldItemCount = _tokenIds.current() -
_itmesSold.current();
        uint256 currentIndex = 0;
        MarketItem[] memory items = new MarketItem[](unSoldItemCount);
        for (uint256 i = 0; i < itemCount; i++) {
            if (idMarketItem[i + 1].owner == address(this)) {
                uint256 currentId = i + 1;
                MarketItem storage currentItem = idMarketItem[currentId];
                items[currentIndex] = currentItem;
                currentIndex += 1;
            }
        }
        return items;
    }
```

Функція fetchMyNFT повертає масив токенів, які належать користувачу, що викликає цю функцію. Вона проходить через всі створені токени і вибирає ті, що мають власника msg.sender (Лістинг 2.10).

Лістинг 2.10 – Функція, яка повертає токени користувача

```
  function fetchMyNFT() public view returns (MarketItem[] memory) {
        uint256 totalCount = _tokenIds.current();
        uint256 itemCount = 0;
        uint256 currentIndex = 0;
        for (uint256 i = 0; i < totalCount; i++) {
            if (idMarketItem[i + 1].owner == msg.sender) {
                itemCount += 1;
            }
        }
        MarketItem[] memory items = new MarketItem[](itemCount);
        for (uint256 i = 0; i < totalCount; i++) {
            if (idMarketItem[i + 1].owner == msg.sender) {
                uint256 currntId = i + 1;
                MarketItem storage currentItem = idMarketItem[currntId];
                items[currentIndex] = currentItem;
                currentIndex += 1;
            }
        }
        return items;
    }
```

Функція fetchItemsListed повертає масив токенів, які були виставлені на продаж користувачем, що викликає цю функцію. Вона проходить через всі створені токени і вибирає ті, що мають продавця msg.sender (Лістинг 2.11).

Лістинг 2.11 – Функція, яка повертає виставлені на продаж токени

```
  function fetchItemsListed() public view returns (MarketItem[] memory)
{
        uint256 totalCount = _tokenIds.current();
        uint256 itemCount = 0;
        uint256 currentIndex = 0;
        for (uint256 i = 0; i < totalCount; i++) {
```

```
            if (idMarketItem[i + 1].seller == msg.sender) {
                itemCount += 1;
            }
        }
        MarketItem[] memory items = new MarketItem[](itemCount);
        for (uint256 i = 0; i < totalCount; i++) {
            if (idMarketItem[i + 1].seller == msg.sender) {
                uint256 currentId = i + 1;
                MarketItem storage currentItem = idMarketItem[currentId];
                items[currentIndex] = currentItem;
                currentIndex += 1;
            }
        }
        return items;
    }
}
```

2.3 Розробка смарт-контракта англійського аукціону

Англійський аукціон – це тип аукціону, де ціна товару або послуги зростає з кожною новою ставкою. Процес починається з оголошення стартової ціни, після чого учасники роблять ставки, перевищуючи поточну найвищу пропозицію. Аукціон завершується, коли ніхто з учасників не хоче запропонувати вищу ставку, і товар продається тому, хто запропонував найвищу ціну.

Англійські аукціони широко використовуються для продажу різноманітних товарів, включаючи антикваріат, мистецтво та нерухомість. У випадку з криптовалютами та NFT, англійські аукціони дозволяють прозоро та справедливо встановлювати ринкову ціну активів.

Смарт-контракт для реалізації англійського аукціону для NFT написано на базі стандарту ERC-721 (Лістинг 2.12). Контракт дозволяє продавцю почати аукціон, приймати ставки, знімати ставки і завершувати аукціон, передаючи NFT переможцю або повертаючи його продавцю в разі відсутності ставок.

Лістинг 2.12 – Смарт-контракт англійського аукціону

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;
interface IERC721 {
    function safeTransferFrom(address from, address to, uint256 tokenId)
        external;
    function transferFrom(address, address, uint256) external;
}
contract EnglishAuction {
    event Start();
    event Bid(address indexed sender, uint256 amount);
    event Withdraw(address indexed bidder, uint256 amount);
    event End(address winner, uint256 amount);
    IERC721 public nft;
    uint256 public nftId;
    address payable public seller;
    uint256 public endAt;
    bool public started;
    bool public ended;
    address public highestBidder;
    uint256 public highestBid;
    mapping(address => uint256) public bids;
    constructor(address _nft, uint256 _nftId, uint256 _startingBid) {
        nft = IERC721(_nft);
        nftId = _nftId;
        seller = payable(msg.sender);
        highestBid = _startingBid;
    }
    function start() external {
        require(!started, "started");
        require(msg.sender == seller, "not seller");
        nft.transferFrom(msg.sender, address(this), nftId);
        started = true;
        endAt = block.timestamp + 7 days;
        emit Start();
    }
    function bid() external payable {
        require(started, "not started");
        require(block.timestamp < endAt, "ended");
```

```
        require(msg.value > highestBid, "value < highest");
        if (highestBidder != address(0)) {
            bids[highestBidder] += highestBid;
        }
        highestBidder = msg.sender;
        highestBid = msg.value;
        emit Bid(msg.sender, msg.value);
    }
    function withdraw() external {
        uint256 bal = bids[msg.sender];
        bids[msg.sender] = 0;
        payable(msg.sender).transfer(bal);
        emit Withdraw(msg.sender, bal);
    }
    function end() external {
        require(started, "not started");
        require(block.timestamp >= endAt, "not ended");
        require(!ended, "ended");
        ended = true;
        if (highestBidder != address(0)) {
            nft.safeTransferFrom(address(this), highestBidder, nftId);
            seller.transfer(highestBid);
        } else {
            nft.safeTransferFrom(address(this), seller, nftId);
        }
        emit End(highestBidder, highestBid);
    }
}
```

Інтерфейс IERC721 визначає дві функції, необхідні для передачі токенів між адресами. Це стандартний інтерфейс для NFT, який забезпечує сумісність з будь-яким контрактом ERC-721.

Основний контракт EnglishAuction включає визначення подій, змінних стану і функцій, необхідних для реалізації англійського аукціону. Події дозволяють відстежувати зміни стану контракту (початок, завершення аукціону, нову ставку та зняття ставки). Змінні стану зберігають інформацію про NFT, продавця, ставки і стан аукціону.

Функція start дозволяє продавцю почати аукціон, передаючи NFT контракту і встановлюючи час закінчення аукціону. Перевіряє, чи аукціон ще не почався і чи викликає функцію продавець.

Функція bid дозволяє учасникам робити ставки. Ставка має бути більшою за поточну найвищу ставку.

Функція withdraw дозволяє учасникам знімати свої ставки, якщо їх ставка була перевищена.

Функція end завершує аукціон, передаючи NFT переможцю або повертаючи його продавцю, якщо не було ставок.

2.4 Розробка смарт-контракта голландського аукціону

Голландський аукціон – це тип аукціону, в якому початкова ціна товару або послуги встановлюється високою і поступово знижується з часом до тих пір, поки хтось з учасників не погодиться купити за поточною ціною. Це протилежність англійському аукціону, де ціна зростає з кожною новою ставкою. В голландському аукціоні товар продається першому учаснику, який погоджується на поточну ціну, тому процес часто відбувається дуже швидко.

Смарт-контракт встановлює початкову ціну, яку знижують із часом до завершення аукціону. Якщо учасник погоджується на поточну ціну, він купує NFT, і аукціон завершується (Лістинг 2.13).

Лістинг 2.13 – Смарт-контракт голландського аукціону

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;
interface IERC721 {
    function transferFrom(address _from, address _to, uint256 _nftId)
        external;
}
contract DutchAuction {
    uint256 private constant DURATION = 7 days;
    IERC721 public immutable nft;
    uint256 public immutable nftId;
    address payable public immutable seller;
    uint256 public immutable startingPrice;
```

```
    uint256 public immutable startAt;
    uint256 public immutable expiresAt;
    uint256 public immutable discountRate;
    constructor(
        uint256 _startingPrice,
        uint256 _discountRate,
        address _nft,
        uint256 _nftId
    ) {
        seller = payable(msg.sender);
        startingPrice = _startingPrice;
        startAt = block.timestamp;
        expiresAt = block.timestamp + DURATION;
        discountRate = _discountRate;
        require(
            _startingPrice >= _discountRate * DURATION, "starting price <
min"
        );
        nft = IERC721(_nft);
        nftId = _nftId;
    }
    function getPrice() public view returns (uint256) {
        uint256 timeElapsed = block.timestamp - startAt;
        uint256 discount = discountRate * timeElapsed;
        return startingPrice - discount;
    }
    function buy() external payable {
        require(block.timestamp < expiresAt, "auction expired");
        uint256 price = getPrice();
        require(msg.value >= price, "ETH < price");
        nft.transferFrom(seller, msg.sender, nftId);
        uint256 refund = msg.value - price;
        if (refund > 0) {
            payable(msg.sender).transfer(refund);
        }
        selfdestruct(seller);
    }
}
```

Інтерфейс IERC721 визначає функції, необхідні для передачі NFT між адресами.

Конструктор ініціалізує контракт, встановлюючи початкові значення для змінних аукціону, таких як початкова ціна, ставка зниження, адреса продавця, адреса NFT і ID NFT.

Функція getPrice обчислює поточну ціну NFT, враховуючи час, що минув з початку аукціону і зниження ціни з кожною секундою.

Функція buy дозволяє покупцям купувати NFT за поточною ціною. Вона перевіряє, чи не закінчився аукціон, і чи надана сума відповідає поточній ціні. Після успішної покупки, NFT передається покупцю, а залишок коштів повертається.

2.5 Оптимізація за допомогою Yul

Для прикладу буде розглянуто оптимізацію коду представленого у лістингу 2.4. Перед оптимізацією розгорнемо контракт у блокчейні та подивимось скільки газу витрачається на його розгортання (Рисунок 2.1). На рисунку видно, що execution cost = 169239.

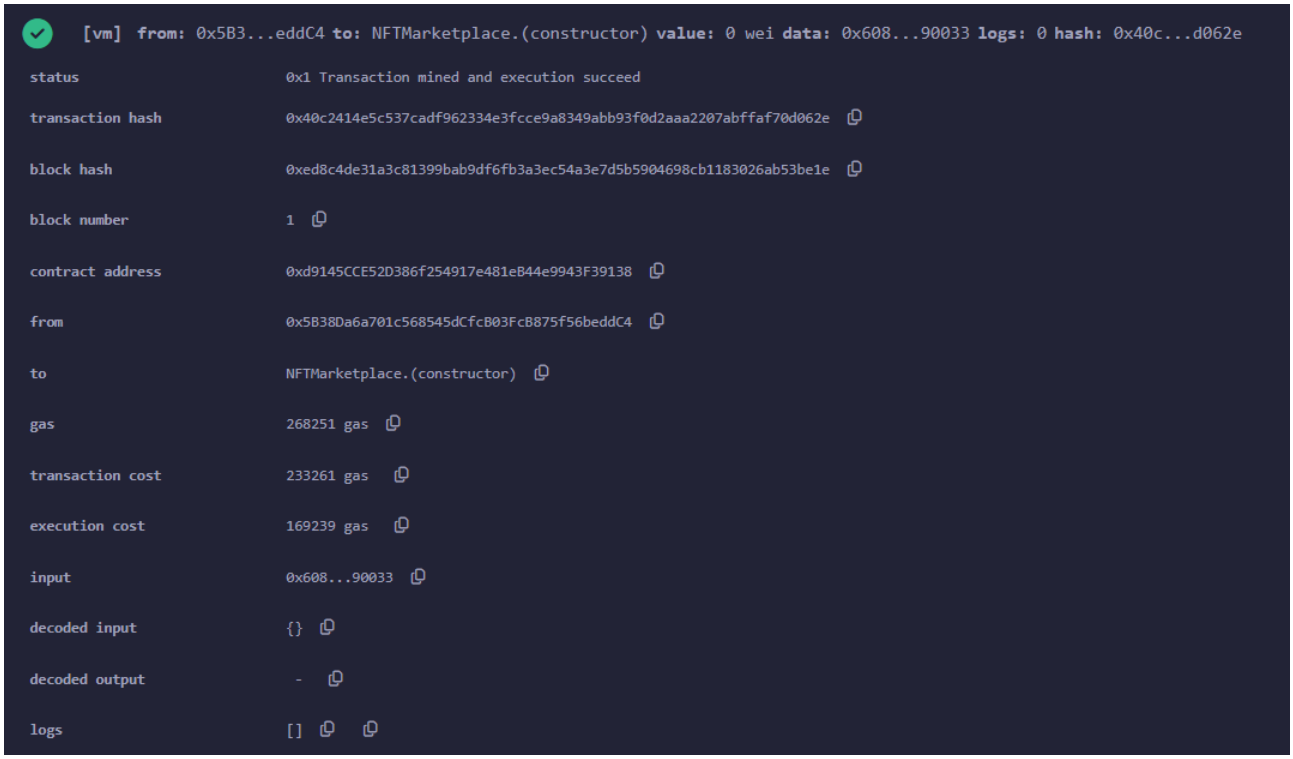

Рисунок 2.1 – Розгортання контракту

На підготовчому етапі створено фунцію, яка повертає представлення рядка у закодованому вигляді та довжину цього рядка:

```
function prep() external pure returns(bytes memory enc, uint len){
    enc = abi.encode("only owner of the marketplace can change the
listing price");
    len = bytes("only owner of the marketplace can change the
listing price").length;
}
```

Після виклику функції було виведено рядок та його довжину (Рисунок 2.2). Для зручності розуміння рядок було поділено на 4 рівні частини (Рисунок 2.3). Число 20 у першому рядку це зсув, яке каже про те де саме рядок починається. У другому рядку 3а це довжина рядка у шістнадцятирічній системі, тобто у десятичній системі це дорівнює 58. Третій та четверний рядок це рядок тексту, який було закодовано.

Рисунок 2.2 – Закодований рядок

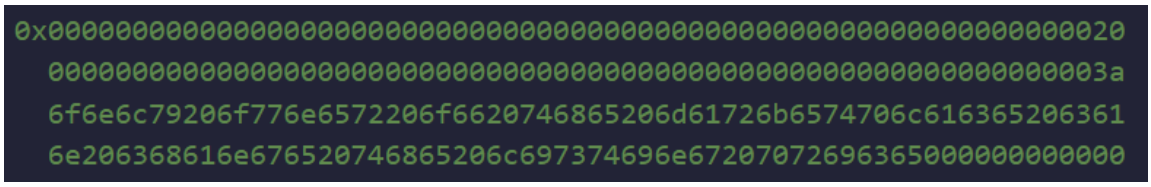

Рисунок 2.3 – Рядок розбитий на чотири частини

Оптимізована функція updateListingPrice має умову if. Спочатку викликається інструкція caller(), яка рівносильна msg.sender. Потім робиться прив’язка до слота де знаходиться owner (owner.slot) і виконується інструкція sload(), тобто з цього слота завантажується відповідне значення. Після чого виконується інструкція sub(), яка віднімає одне значення від іншого (Лістинг 2.14). Якщо результатом перевірки буде нуль, відкату транзакції не станеться.

Всередині умови виконується зберігання у пам’яті. У нульовий слот завантажено обов’язковий відступ 0x20, а у слот 0x20 зберігаємо довжину рядка. Для зберігання значення рядка використовуємо два слоти, 0x40 та 0x60, тому що в один одному слоті (256 біт) весь рядок не поміститься. Далі виконується відкат транзакції інструкцією revert(), яка приймає два значення: слот з якого починати та яким закінчити, не включаючи його. Візуальне представлення пам’яті та даних, що зберігаються, наведено на рисунку 2.4.

Хоча нульовий слот пам’яті зарезервовано для обчислення хешу, а наступний для free-memory pointer (покажчик вільної пам’яті), за допомогою Yul ми можемо їх зайняти, оскільки точно відомо що далі у коді ці службові речі не будуть використовуватись. Це дозволяє зменшити витрати газу при розгортанні контратку та при створенні транзакцій.

Лістинг 2.14 – Оптимізований смарт-контракт змінення вартості NFT

```
contract NFTMarketplaceYul {
    uint256 listingPrice = 0.00025 ether;
    address payable owner;
    constructor() {
        owner = payable(msg.sender);
    }
    function updateListingPrice(uint256 _listingPrice)
        public
        payable
```

```
    {
        assembly{
            if sub(caller(), sload(owner.slot)){
                mstore(0x00, 0x20)
                mstore(0x20, 0x3a)
                mstore(0x40,
0x6f6e6c79206f776e6572206f6620746865206d61726b6574706c616365206361)
                mstore(0x60,
0x6e206368616e676520746865206c697374696e67207072696365000000000000)
                revert(0x00, 0x80)
            }
        }
        listingPrice = _listingPrice;
    }
    function getListingPrice() public view returns (uint256) {
        return listingPrice;
    }}
```

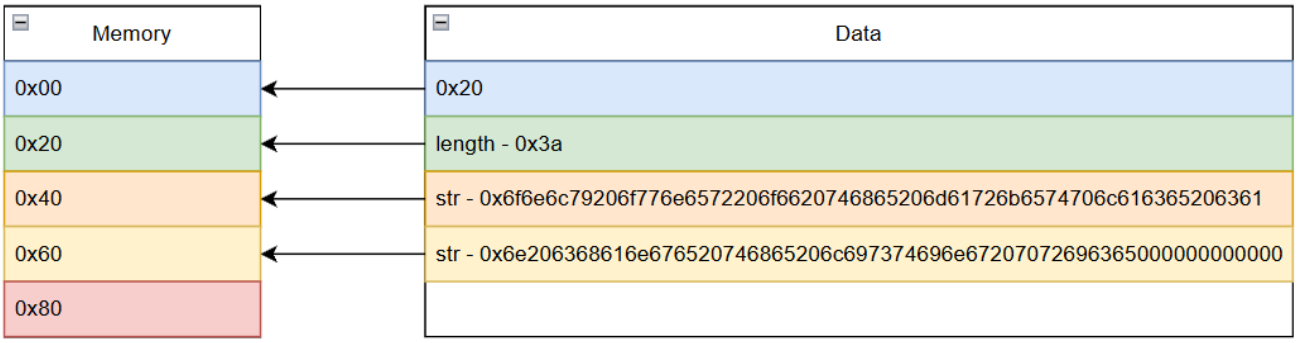

Рисунок 2.4 – Візуальне представлення пам’яті

Результат розгортання оптимізованого контракту представлено на рисунку 2.5. Після оптимізації, вартість розгортання зменшилась з 169239 до 128203 gas.

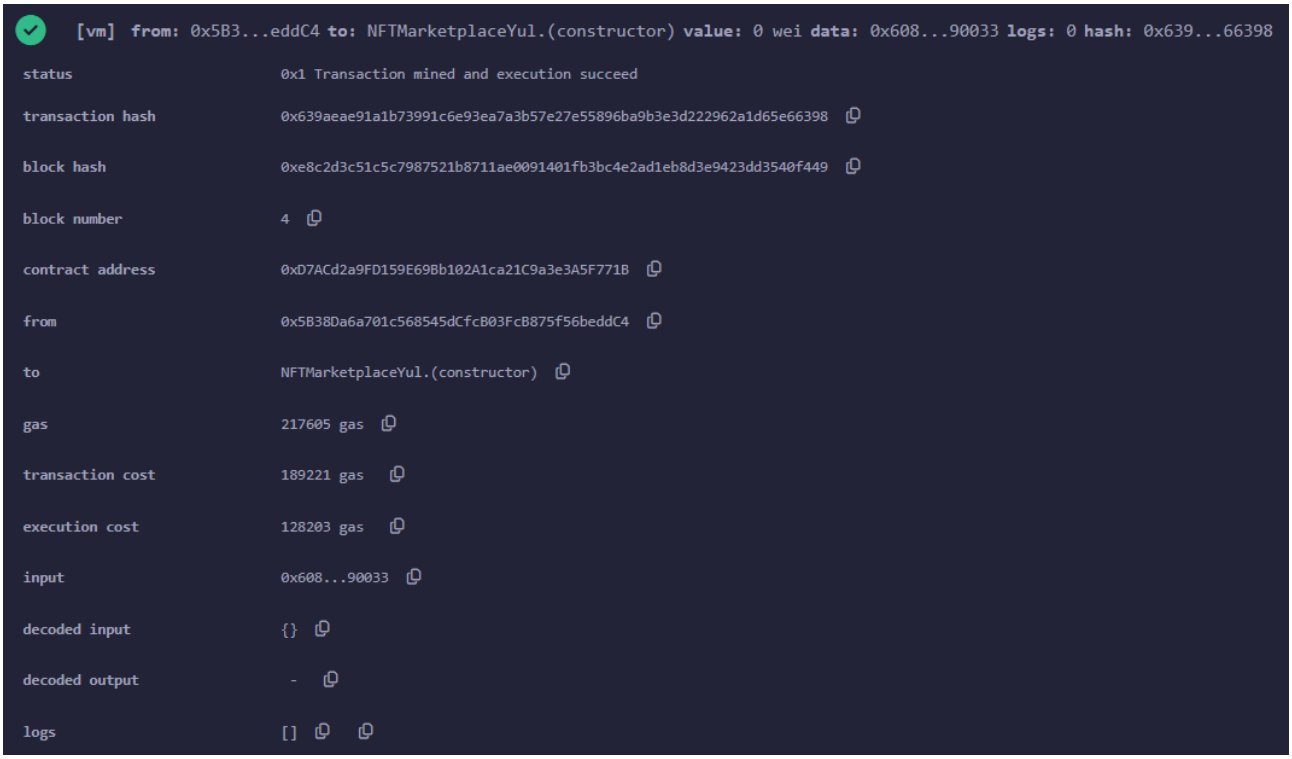

Рисунок 2.5 – Розгортання отпимізованого контракту

2.6 Розгортання контракту

Для розгортання контракту використовується інструмент hardhat. Завантаження hardhat було здійснено за допомогою менеджера пакетів, командою npx hardhat init. Після цього було змінено стандартний скрипт для розгортання на свій, код якого представлено у лістингу 2.15.

Лістинг 2.15 – Скрипт для розгортання контракту

```
const hre = require("hardhat");
async function main() {
  const NFTMarketplace = await
hre.ethers.getContractFactory("NFTMarketplace");
  const nftMarketplace = await NFTMarketplace.deploy();
  await nftMarketplace.deployed();
  console.log(` deployed contract Address ${nftMarketplace.address}`);
}
main().catch((error) => {
  console.error(error);
  process.exitCode = 1;});
```

Скрипт написаний на JavaScript і розгортає смарт-контракт у блокчейн мережі. Перший рядок імпортує об’єкт hre (Hardhat Runtime Environment), який надає доступ до всіх функцій та утиліт Hardhat. Визначається асинхронна функція main, яка буде виконувати основні операції по розгортанню смарт-контракту. Використовується метод getContractFactory з бібліотеки ethers (інтегрованої з Hardhat) для отримання фабрики контрактів NFTMarketplace. Фабрика контрактів – це об’єкт, який використовується для створення нових екземплярів смарт-контрактів. Викликається метод deploy для розгортання нового екземпляру смарт-контракту NFTMarketplace на блокчейн. Цей процес створює транзакцію розгортання контракту. Очікується завершення транзакції розгортання контракту. Метод deployed повертає проміс, який повідомляє коли контракт успішно розгорнуто. Після успішного розгортання виводиться адреса нового контракту в консоль. Викликається функція main і обробляються будь-які помилки, які можуть виникнути під час виконання. Якщо виникає помилка, вона виводиться в консоль, і процес завершується з кодом помилки 1.

Перед розгортанням контракту треба запустити локальну ноду командою npx hardhat node (Рисунок 2.6). Також автоматично створюються гаманці з тестовим балансом.

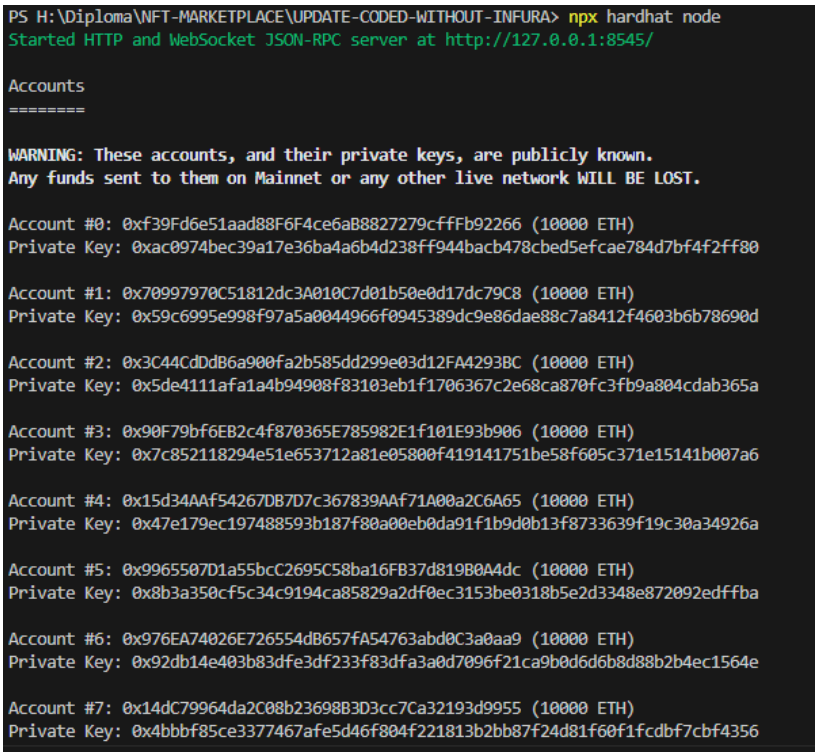

Рисунок 2.6 – Запуск ноди

Для того щоб локально розгорнути контракт потрібно ввести команду npx hardhat run scripts/deploy.js --network localhost (Рисунок 2.7).

```
PS H:\Diploma\NFT-MARKETPLACE\UPDATE-CODED-WITHOUT-INFURA> npx hardhat run scripts/deploy.js --networ
k localhost
 deployed contract Address 0x5FbDB2315678afecb367f032d93F642f64180aa3
PS H:\Diploma\NFT-MARKETPLACE\UPDATE-CODED-WITHOUT-INFURA>
```

Рисунок 2.7 – Розгортання смарт-контракта

Все працює без помилок, тому можна розгортати контракт не локально, а на Ethereum. Було обрано тестову мережу Ethereum Sepolia. Sepolia – це тестова мережа (testnet) Ethereum, яка використовується для випробувань і розробки смарт-контрактів перед їх розгортанням у головній мережі (mainnet). Вона дозволяє розробникам тестувати свої контракти та додатки в умовах, максимально наближених до реальних, без витрат на ефір (ETH) і без ризику втратити кошти через помилки або збої.

Переходимо на сайт Alchemy та створюємо новий проєкт (Рисунок 2.8). Відкривається вікно для моніторингу та аналітики транзакцій (Рисунок 2.9). Натискаємо на кнопку API key і копіюємо ключ.

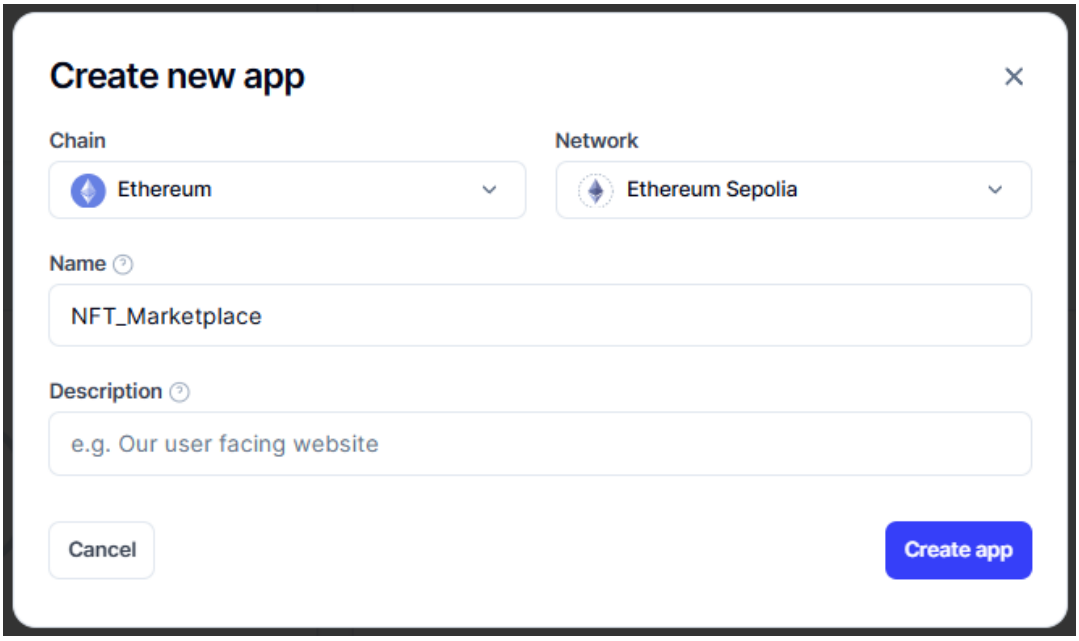

Рисунок 2.8 – Створення проєкту в Alchemy

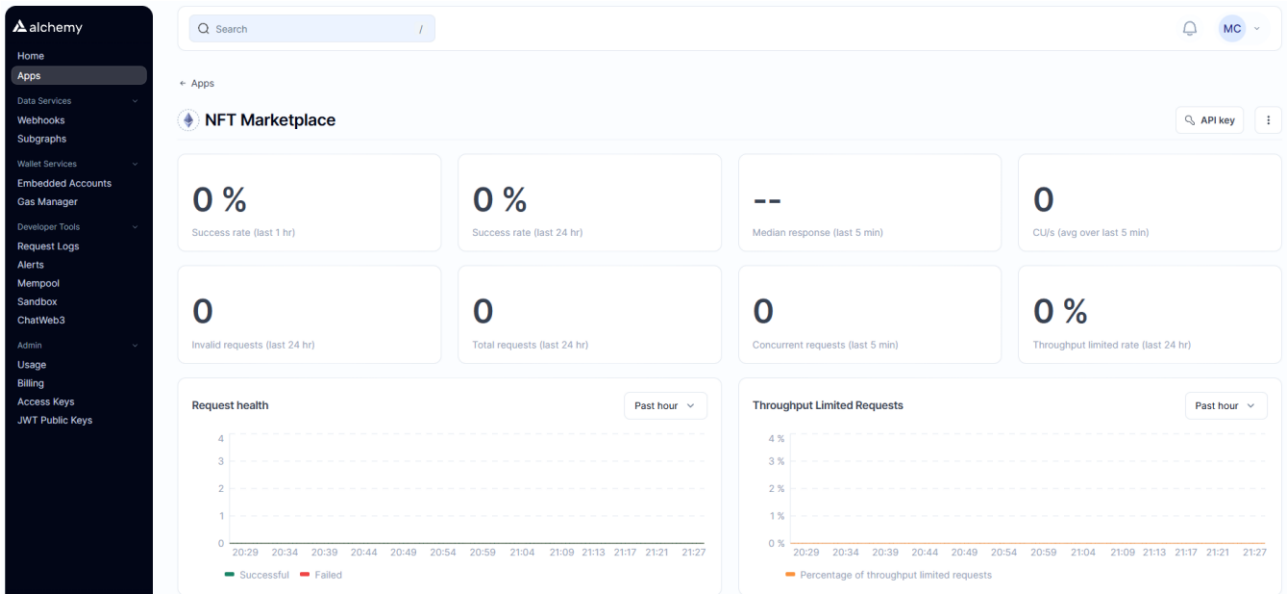

Рисунок 2.9 – Сторінка для моніторингу та аналітики

Це було потрібно для подальшого налаштування. Після цього було змінено файл налаштування hardhat (hardhat.config.js). Код представлено у лістингу 2.16.

Лістинг 2.16 – Налаштування hardhat

```
require("@nomicfoundation/hardhat-toolbox");
const NEXT_PUBLIC_PRIVATE_KEY =
"cbdcd3edd365d844497b3a8edd8dcd9c6d5788ffdb2f7f1f4567975b08cdd";
const NEXT_PUBLIC_ETH_SEPOLIA_RPC = "https://eth-
sepolia.g.alchemy.com/v2/0dz99eYhGfDu0JNKl1lRlx47iHlHR28S"
/** @type import('hardhat/config').HardhatUserConfig */
module.exports = {
  solidity: "0.8.17",
  defaultNetwork: "eth",
  networks: {
    hardhat: {},
    eth_sepolia: {
      url: NEXT_PUBLIC_ETH_SEPOLIA_RPC,
      accounts: [`0x${NEXT_PUBLIC_PRIVATE_KEY}`],
    },
  },
};
```

У коді вказуються приватний ключ від гаманця Metamask, API ключ від Alchemy та налаштування мережі Sepolia.

В консоль вводимо команду hardhat run scripts/deploy.js --network eth_sepolia і отримуємо адресу смарт-контракта - 0xa5853fFDAA94528a152fF171A6f2875905edaDE0 (Рисунок 2.10).

```
PS H:\Diploma\NFT-MARKETPLACE\UPDATE-CODED-WITHOUT-INFURA> npx hardhat run scripts/deploy.js --network eth_sepolia
Compiled 1 Solidity file successfully
 deployed contract Address 0xa5853fFDAA94528a152fF171A6f2875905edaDE0
PS H:\Diploma\NFT-MARKETPLACE\UPDATE-CODED-WITHOUT-INFURA>
```

Рисунок 2.10 – Розгортання смарт-контракта у мережі Sepolia

На сайті sepolia.etherscan.io можна ввести адресу контракта та побачити усі транзакції здійснені з ним (Рисунок 2.11).

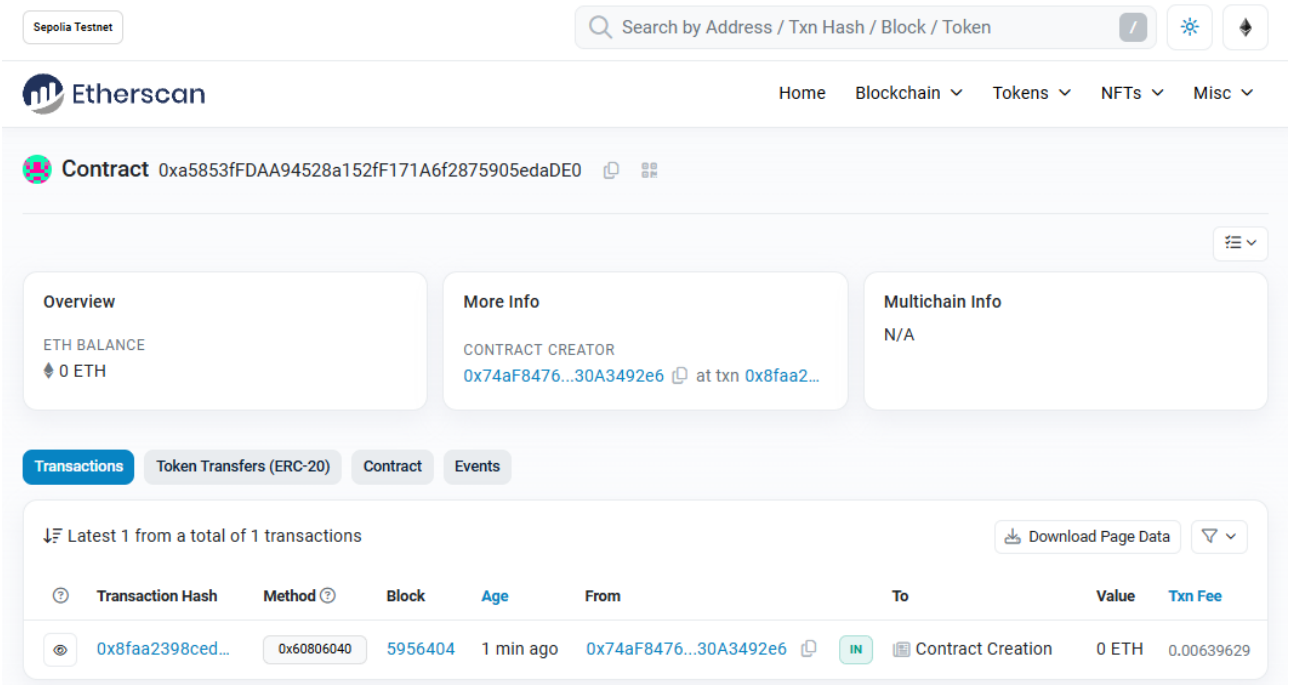

Рисунок 2.11 – Смарт-контракт на etherscan

3 ПРОЕКТУВАННЯ ТА РОЗРОБКА ВЕБЗАСТОСУНКУ NFT-МАРКЕТПЛЕЙСУ

3.1 Огляд фреймворку для розробки

Для створення вебзастосунку NFT-маркетплейсу було вирішено використовувати наступні технології:

– React – бібліотека для створення інтерфейсів користувача на основі компонентів;

– CSS – мова стилів для оформлення зовнішнього вигляду вебсторінок;

– Next.js – фреймворк для серверного рендерингу і статичної генерації сторінок на основі React;

– Ethers.js – бібліотека для взаємодії з блокчейном та смарт-контрактами;

– Web3modal – бібліотека для підключення до криптовалютних гаманців;

– Axios – бібліотека для виконання HTTP-запитів до API;

– MetaMask – криптовалютний гаманець і розширення для браузера для взаємодії з Ethereum;

– Pinata – сервіс для зберігання файлів на IPFS та управління ними.

3.1.1 Фреймворк React

React – це популярна JavaScript-бібліотека для створення інтерфейсів користувача, розроблена та підтримувана компанією Facebook. Вона дозволяє розробникам створювати динамічні та інтерактивні веб-додатки з використанням компонентного підходу [23, 24].

Основні особливості React включають:

– компонентний підхід: React використовує компонентну архітектуру, що дозволяє розробникам розділяти інтерфейс на незалежні, повторно використовувані компоненти. Кожен компонент відповідає за власну частину UI і може бути використаний у різних частинах додатку;

– віртуальний DOM: React використовує віртуальний DOM, що дозволяє оптимізувати оновлення інтерфейсу. При зміні стану додатку React оновлює тільки ті частини DOM, які змінилися, що підвищує продуктивність;

– jsx: React використовує JSX - розширення JavaScript, яке дозволяє писати HTML-подібний код всередині JavaScript. Це робить код більш читабельним і зручним для розробників;

– односторонній потік даних: React реалізує односторонній потік даних, це означає, що дані передаються від батьківських компонентів до дочірніх через властивості (props). Це допомагає забезпечити передбачуваність і легкість в налагодженні додатку.

3.1.2 Мова каскадних таблиць стилів (CSS)

CSS (Cascading Style Sheets) – це мова стилів, яка використовується для опису вигляду та форматування HTML-документів. CSS дозволяє розробникам відокремлювати вміст веб-сторінки від її стилю, забезпечуючи гнучкість та зручність у створенні веб-дизайнів [25].

Основні особливості CSS включають:

– стилізація елементів: CSS дозволяє змінювати вигляд HTML-елементів, включаючи кольори, шрифти, розміри, відступи та багато іншого;

– каскадність та наслідування: CSS має каскадну природу, що означає, що стилі можуть наслідуватися від батьківських елементів до дочірніх. Це дозволяє ефективно застосовувати стилі на різних рівнях документу та уникати дублювання коду;

– адаптивний дизайн: CSS підтримує адаптивний дизайн через медіа-запити, що дозволяє створювати інтерфейси, які виглядають добре на різних пристроях та екранах.

3.1.3 Next.js

Next.js – це фреймворк для React, який дозволяє створювати серверно-рендеровані та статичні веб-додатки. Розроблений компанією Vercel, Next.js додає додаткові функціональні можливості до React, роблячи його більш потужним та зручним для розробки складних веб-додатків [26, 27].

Основні особливості Next.js включають:

– серверний рендеринг (SSR): Next.js підтримує серверний рендеринг, що дозволяє рендерити React-компоненти на сервері перед їх відправкою на клієнт. Це покращує продуктивність і пошукову оптимізвцію (SEO), оскільки сторінки відображаються швидше і можуть бути індексовані пошуковими системами;

– статичне генерування (SSG): Next.js підтримує статичне генерування сторінок, що дозволяє створювати повністю статичні веб-сайти;

– автоматичний розподіл коду: Це означає, що тільки необхідні частини коду завантажуються на клієнтську сторону, покращуючи швидкість завантаження сторінок;

– оптимізація зображень: Next.js має вбудовану оптимізацію зображень, що дозволяє автоматично оптимізувати завантаження та відображення зображень для покращення продуктивності.

3.1.4 Ethers.js

Ethers.js – це бібліотека JavaScript для взаємодії з мережею Ethereum та розробки додатків, які працюють з блокчейном Ethereum [28].

Основні особливості Ethers.js включають:

– відправлення та отримання транзакцій: Бібліотека надає інтерфейс для створення та підписання транзакцій, а також для відправлення їх на мережу Ethereum для обробки;

– керування смарт-контрактами: Ethers.js дозволяє взаємодіяти зі смарт-контрактами на Ethereum, відправляти їм транзакції та викликати їхні функції;

– підтримка мережевих подій (events): Ethers.js надає зручний інтерфейс для підписки на події, що відбуваються в мережі Ethereum, такі як підтвердження транзакцій, зміни в смарт-контрактах тощо.

3.1.5 Web3modal

Web3modal – це бібліотека JavaScript, яка дозволяє легко і швидко додавати у свої додатки підтримку різних провайдерів Web3 (MetaMask, WalletConnect, Fortmatic, Portis, Torus та інші) для взаємодії з Ethereum та іншими блокчейнами [29].

Основні особливості Web3modal включають:

– підтримка різних провайдерів: Бібліотека підтримує багато різних провайдерів Web3, що дозволяє користувачам обирати найзручніший для них спосіб взаємодії з додатком;

– автоматичне виявлення доступних провайдерів: Web3modal автоматично виявляє доступні провайдери Web3 та надає користувачам можливість вибору;

– підтримка безпеки: Web3modal підтримує ряд безпекових практик, таких як перевірка підпису повідомлення для підтвердження ідентифікації користувача.

3.1.6 Axios

Axios – це бібліотека для виконання HTTP-запитів у середовищі Node.js та веб-браузерах. Вона надає зручний і простий спосіб взаємодії з веб-серверами для отримання даних, відправлення форм і виконання інших HTTP-дій [30].

Основні особливості Axios включають:

– підтримка перехоплення запитів та відповідей: Axios дозволяє перехоплювати та обробляти запити та відповіді для виконання додаткової логіки, наприклад, для обробки помилок або авторизації;

– підтримка відправки файлів: Бібліотека дозволяє відправляти файли на сервер у складі запиту;

– підтримка встановлення заголовків (headers): Axios дозволяє додавати та налаштовувати HTTP-заголовки для кожного запиту;

– можливість відправляти дані у форматі JSON: Axios дозволяє відправляти дані у форматі JSON для обміну даними з сервером.

3.1.7 MetaMask

MetaMask – це веб-розширення та мобільний додаток, які дозволяють користувачам створювати та керувати гаманцями для криптовалют та взаємодіяти з додатками, побудованими на блокчейні Ethereum [31].

Основні особливості MetaMask включають:

– взаємодія з додатками на блокчейні: MetaMask надає інтерфейс для взаємодії з додатками, побудованими на EVM-сумісних блокчейнах;

– безпека та приватність: MetaMask забезпечує високий рівень безпеки та приватності, зберігаючи приватні ключі користувачів локально на їх пристроях;

– налаштування мереж: Користувачі можуть додавати власні мережі, вказуючи їхні RPC-адреси (Remote procedure call), ідентифікатори, назви та символи валюти. Це дозволяє підключатися до тестових мереж або приватних блокчейнів;

– налаштування токенів: Крім токенів, які включені за замовчуванням, користувачі можуть додавати власні токени за допомогою адреси контракту токена, назви та символу. Це дозволяє відстежувати та взаємодіяти з різними токенами на різних мережах Ethereum.

3.1.8 Pinata

Pinata – це платформа для зберігання та управління IPFS (InterPlanetary File System) об’єктами. IPFS – це протокол для розподіленого зберігання та передачі гіпермедіа-вмісту через мережу [32].

Основні особливості Pinata включають:

– зберігання файлів на IPFS: Pinata дозволяє користувачам зберігати файли на мережі IPFS, що забезпечує децентралізоване та надійне зберігання;

– управління об’єктами IPFS: Pinata надає інтерфейс для завантаження, керування та взаємодії з об’єктами, збереженими на IPFS;

– моніторинг та аналітика: Pinata надає засоби для моніторингу та аналітики використання об’єктів на IPFS.

3.2 Функціонал вебзастосунку NFT-маркетплейсу

Функціонал вебзастосунку NFT-маркетплейсу можна представити за допомогою Use Case діаграми, яка ілюструє основні функції системи. Діаграма відображає взаємодію між користувачем (актором) та системою [33].

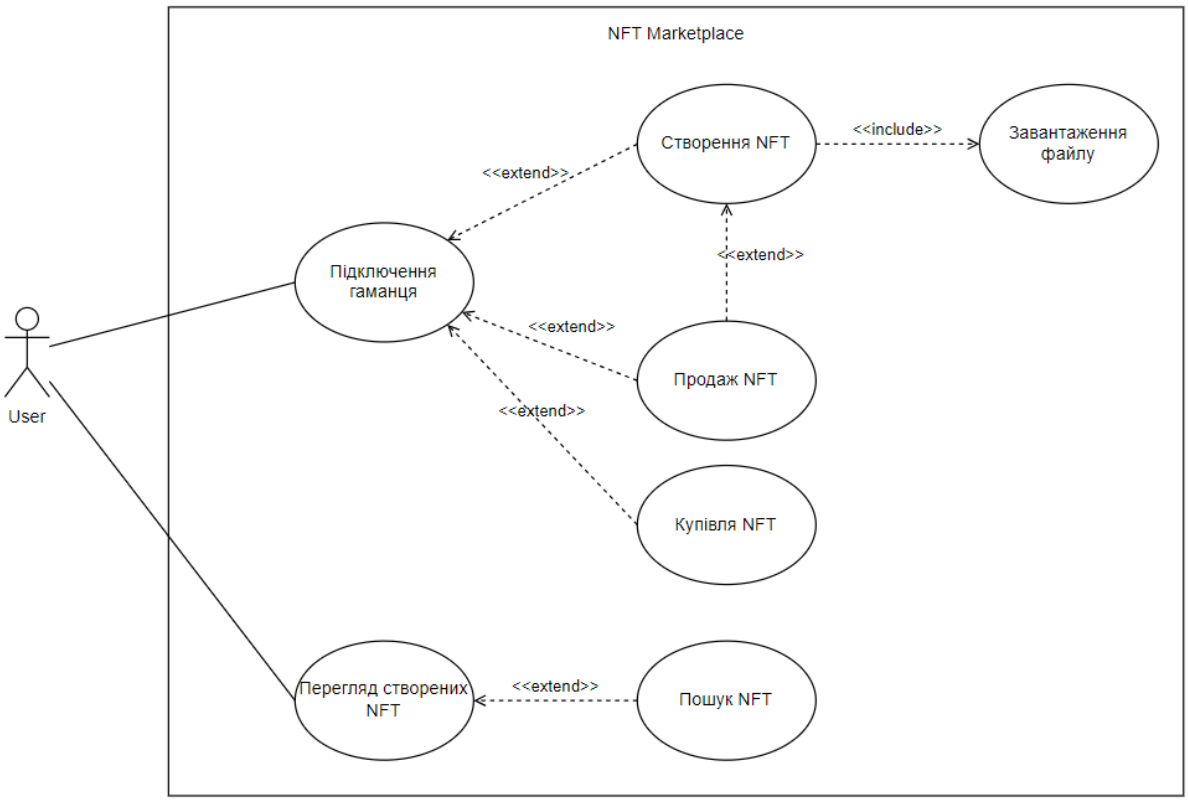

Рисунок 3.1 – Діаграма прецедентів NFT-маркетплейсу

На рисунку 3.1 зображено діаграму прецедентів. Без авторизації користувач має можливість переглядати виставлені на продаж NFT та виконувати пошук по назвам. Авторизація виконується за допомогою криптогаманця. Після підключення гаманця користувач може купувати, продавати та створювати власні NFT. При створенні NFT на сайті, вона автоматично виставляється на продаж по заданій ціні. Створення включає в себе завантаження файлу та опис.

3.3 Діаграми послідовностей дій (Sequence Diagram) вебзастосунку

Діаграми послідовностей дій – це графічне представлення взаємодії між об’єктами у часі. Вони показують, як об’єкти обмінюються повідомленнями для виконання певної функції або процесу в системі, відображаючи порядок і час цих взаємодій [34].

На рисунку 3.2 представлено діаграму послідовності дій створення NFT. Користувач завантажує файл на сторінці створення NFT. Файл зберігається у Pinata IPFS сервісі, а сервіс повертає посилання на файл. Користувач натискає

кнопку створити нфт, сайт викликає Metamask для підпису транзакції та сплати комісії. Користувач натискає на кнопку підпису у гаманці. Гаманець через RPC відправляє транзакцію до блокчейну, створюючи запис про NFT та його власника у блокчейні. Alchemy RPC повертає результат транзакції у Metamask, а гаманець відправляє інформацію користувачу.

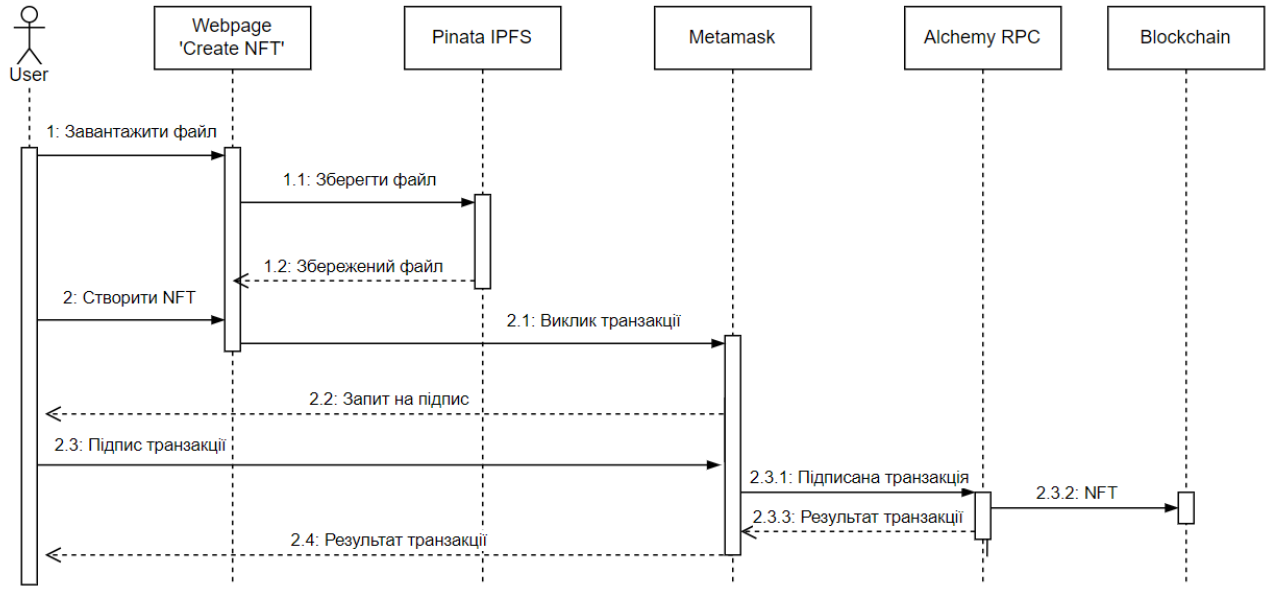

Рисунок 3.2 – Sequence Diagram створення NFT

На рисунку 3.3 представлено діаграму послідовності дій виставлення NFT на продаж. Продавець натискає кнопку «Sell NFT» на сторінці огляду своєї NFT. Сайт викликає Metamask для підпису транзакції та сплати комісії. Продавець натискає на кнопку підпису у гаманці. Гаманець за допомогою RPC взаємодіє зі смарт-контрактом і відправляє йому NFT та дані про NFT. Смарт-контракт повертає результат виконання контракту, а RPC відправляє дані про транзакцію у блокчейн. Після цього Alchemy RPC повертає результат транзакції у Metamask, а гаманець відправляє інформацію продавцю.

На рисунку 3.4 представлено діаграму послідовності дій купівлі NFT. Покупець натискає кнопку «Buy NFT» на сторінці огляду NFT. Сайт викликає Metamask для підпису транзакції сплати комісії та зняття плати за NFT. Покупець натискає на кнопку підпису у гаманці. Гаманець за допомогою RPC взаємодіє зі смарт-контрактом та ініціює виконання зобов’язань контракта. Смарт-контракт відправляє криптовалюту за NFT продавцю, NFT відправляє

покупцю та зберігає дані про транзакції у блокчейні. Alchemy RPC повертає результат транзакції у Metamask, а гаманець відправляє інформацію покупцю.

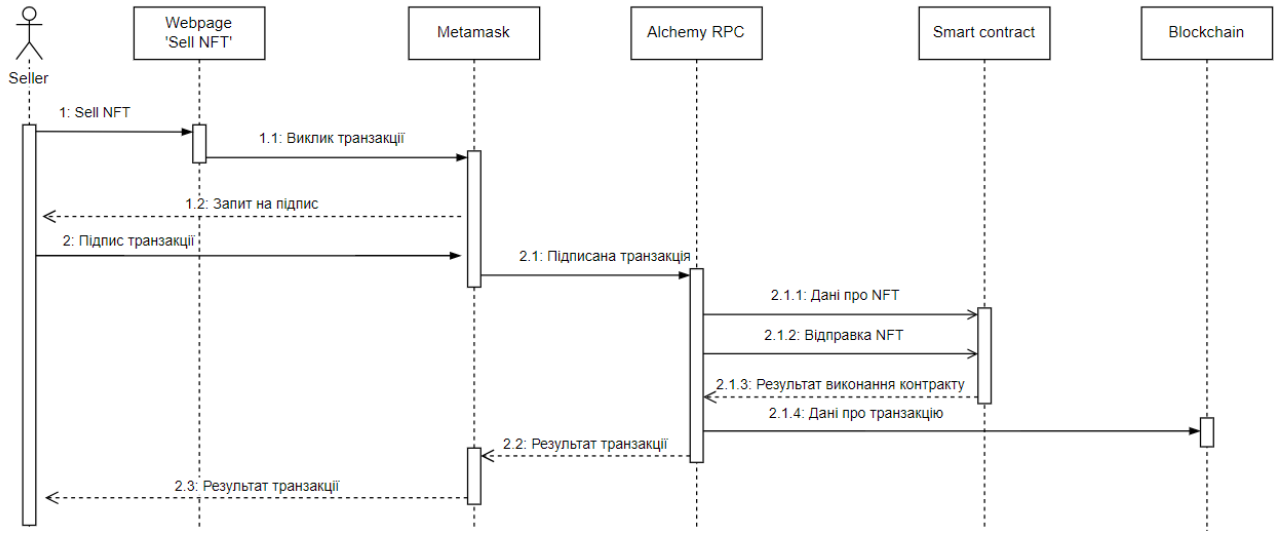

Рисунок 3.3 – Sequence Diagram виставлення NFT на продаж

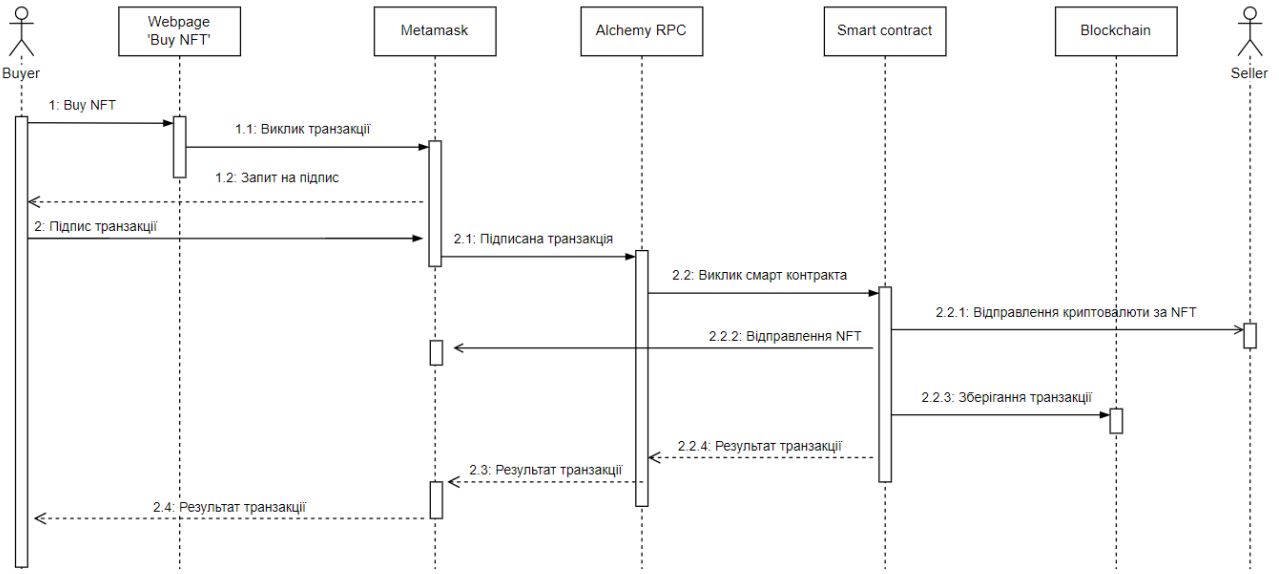

Рисунок 3.4 – Sequence Diagram купівлі NFT

3.4 Розробка вебсайту для взаємодії зі смарт-контрактом на блокчейні

Дерево файлів проекту містить основні папки та файли з кодом додатку, конфігураційні файли для різних середовищ розробки та файли з налаштуваннями. Всі файли та папки відображені на рисунку 3.5.

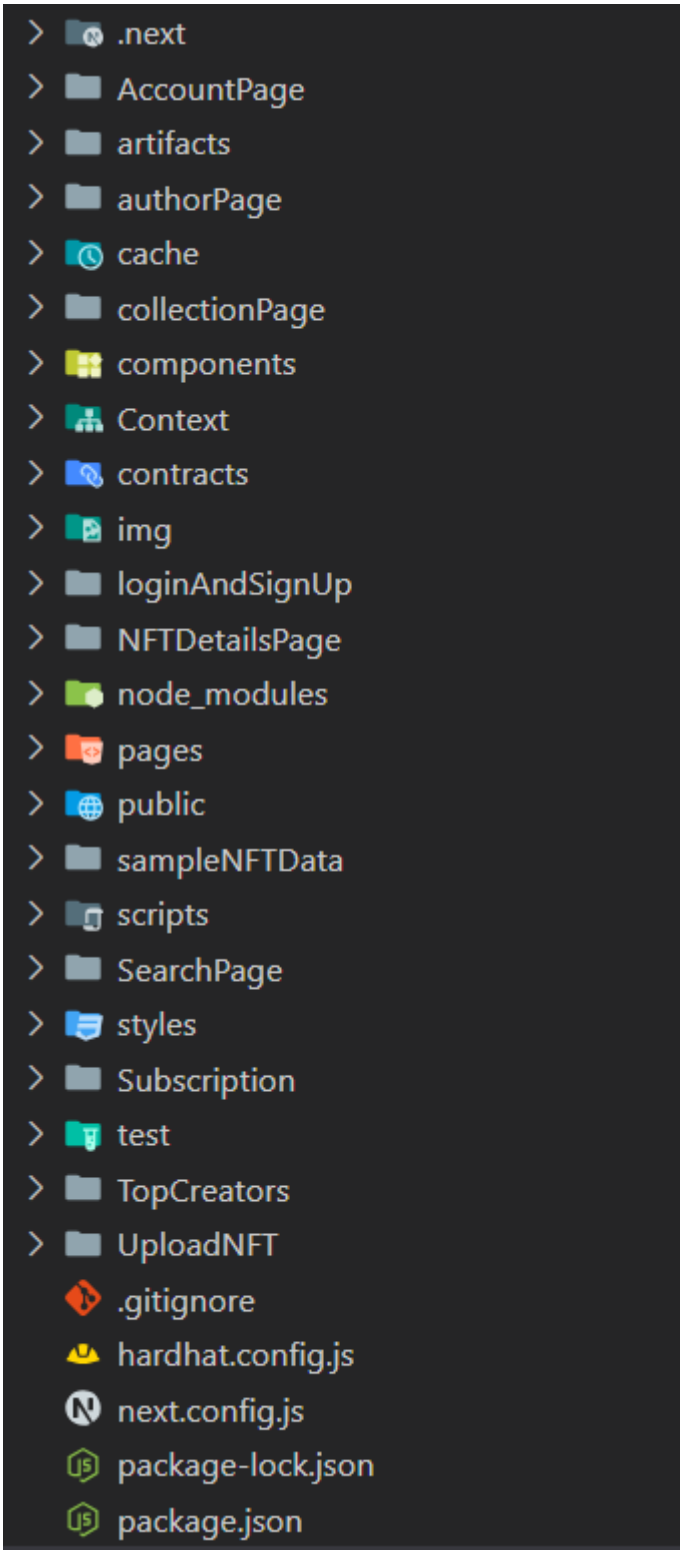

Рисунок 3.5 – Дерево файлів

Опис кожної папки та її вмісту:

– .next: Ця папка генерується автоматично під час компіляції додатку Next.js. Вона містить кешовані файли та результати компіляції, необхідні для швидкого рендерингу сторінок;

– AccountPage: Папка містить файли, що відповідають за відображення та функціонал сторінки акаунту користувача;

– artifacts: Зберігає результати компіляції смарт-контрактів. Згенеровані за допомогою Hardhat. Ці файли використовуються для взаємодії з контрактами;

– authorPage: Містить файли, пов'язані зі сторінкою автора, де відображається інформація про автора та його NFT;

– cache: Використовується для зберігання кешованих даних смарт-контрактів, що прискорюють роботу додатку;

– collectionPage: Файли для відображення колекцій NFT;

– components: Папка містить компоненти коду, які використовуються у різних частинах додатку;

– Context: Містить контекст для управління станом додатку за допомогою React Context API;

– contracts: Папка для смарт-контрактів, написаних на Solidity. Тут знаходяться файли контрактів, які потім компілюються та деплояться на блокчейн;

– img: Містить зображення, що використовуються у додатку;

– loginAndSignUp: Файли, відповідальні за сторінки входу та реєстрації користувачів;

– NFTDetailsPage: Містить файли для відображення деталей конкретного NFT, включаючи його зображення, опис та іншу інформацію;

– node_modules: Папка, де зберігаються всі залежності та пакети, встановлені через npm;

– pages: Основна папка з файлами сторінок Next.js. Кожен файл у цій папці відповідає за окрему сторінку додатку;

– public: Зберігає статичні файли, такі як зображення та шрифти, до яких можна отримати доступ напряму з браузера;

– scripts: Папка для скрипту, який розгортає смарт-контракти;

– SearchPage: Файли для реалізації сторінки пошуку, де користувачі можуть шукати NFT;

– sampleNFTData: Містить приклади даних для демонстрації NFT;

– styles: Папка для файлів стилів CSS, що використовуються для оформлення сторінок з папки pages;

– Subscription: Файли, що відповідають за функціонал підписки користувачів;

– test: Папка яку автоматично створив hardhat для тестів, що перевіряють коректність роботи смарт-контрактів;

– TopCreators: Містить файли для відображення інформації про топових творців на платформі, які заробили найбільше ETH;

– UploadNFT: Файли, відповідальні за функціонал завантаження нових NFT;

– hardhat.config.js: Конфігураційний файл для інструменту Hardhat, що використовується для розробки, тестування та деплою смарт-контрактів;

– next.config.js: Конфігураційний файл для Next.js, який налаштовує параметри збірки та поведінки додатку;

– package-lock.json: Файл, що фіксує версії встановлених залежностей для забезпечення відтворюваності збірок;

– package.json: Основний файл конфігурації для проекту Node.js, де зазначені залежності, скрипти та метадані проекту.

Нижче розглянуто основний файл NFTMarketplaceContext.js, який знаходиться у папці Context та реалізує контекст для NFT-маркетплейсу, що надає функції для взаємодії з блокчейн-мережею та смарт-контрактами. Цей файл забезпечує такі можливості: підключення гаманця, створення NFT, купівля NFT, завантаження даних на IPFS, а також перевірка стану підключення. Він використовує бібліотеки React, Web3Modal, ethers.js та axios (Лістинг 3.1).

Лістинг 3.1 – Імпорт необхідних бібліотек та констант

```
import React, { useState, useEffect } from "react";
import Web3Modal from "web3modal";
import ethers from "ethers";
import useRouter from "next/router";
import axios from "axios";
import {
  NFTMarketplaceAddress,
  NFTMarketplaceABI,
```

```
  transferFundsAddress,
  transferFundsABI,
} from "./constants";
```

Ці імпортовані бібліотеки забезпечують основні функціональні можливості для роботи з React-компонентами, підключенням до крипто-гаманців, взаємодії зі смарт-контрактами та роботи з маршрутизацією в Next.js. Константи містять адреси та ABI (Application Binary Interface) смарт-контрактів, які необхідні для їх викликів.

Лістинг 3.2 – Функція для отримання контракту

```
const fetchContract = (signerOrProvider) =>
  new ethers.Contract(
    NFTMarketplaceAddress,
    NFTMarketplaceABI,
    signerOrProvider
  );
```

Функція у лістингу 3.2 створює новий екземпляр контракту, використовуючи підписувача або провайдера. Вона приймає параметр signerOrProvider, який може бути або підписувачем (signer), або провайдером (provider), і повертає об'єкт контракту для взаємодії з ним.

Лістинг 3.3 – Функція для підключення до смарт-контракту

```
const connectingWithSmartContract = async () => {
  try {
    const web3Modal = new Web3Modal();
    const connection = await web3Modal.connect();
    const provider = new ethers.providers.Web3Provider(connection);
    const signer = provider.getSigner();
    const contract = fetchContract(signer);
    return contract;
  } catch (error) {
    console.log("Something went wrong while connecting with contract",
error);
  }
};
```

У лістингу 3.3 представлено асинхронну функцію, яка підключається до смарт-контракту, використовуючи Web3Modal для підключення гаманця користувача. Після успішного підключення вона створює провайдера та підписувача, а потім повертає екземпляр контракту для подальшої взаємодії.

Лістинг 3.4 – Створення контексту та провайдера контексту

```
export const NFTMarketplaceContext = React.createContext();
export const NFTMarketplaceProvider = ({ children }) => {
  const titleData = "Discover, collect, and sell NFTs";
  // Визначення станів
  const [error, setError] = useState("");
  const [openError, setOpenError] = useState(false);
  const [currentAccount, setCurrentAccount] = useState("");
  const [accountBalance, setAccountBalance] = useState("");
  const router = useRouter();
```

Перший рядок у лістингу 3.4 створює контекст для спільного використання стану та функцій у всьому додатку, що дозволяє різним компонентам взаємодіяти з цим контекстом. Далі провайдер створює контекст, який надає необхідні функції та стан всім дочірнім компонентам. Він визначає стани, що використовуються для зберігання помилок, відкритого стану помилки, поточного акаунта та балансу акаунта.

Лістинг 3.5 – Перевірка підключення гаманця

```
const checkIfWalletConnected = async () => {
  try {
    if (!window.ethereum)
      return setOpenError(true), setError("Install MetaMask");
    const accounts = await window.ethereum.request({
      method: "eth_accounts",
    });
    if (accounts.length) {
      setCurrentAccount(accounts[0]);
    } else {
      console.log("No account");
    }
    const provider = new ethers.providers.Web3Provider(window.ethereum);
```

```
    const getBalance = await provider.getBalance(accounts[0]);
    const bal = ethers.utils.formatEther(getBalance);
    setAccountBalance(bal);
  } catch (error) {
    console.log("not connected");
  }
};
useEffect(() => {
  checkIfWalletConnected();
}, []);
```

Функція checkIfWalletConnected перевіряє, чи підключений гаманець користувача до додатку. Якщо MetaMask не встановлено, вона встановлює стан помилки. В іншому випадку вона отримує акаунти та їх баланс, зберігаючи їх у стані. Використання useEffect забезпечує виконання цієї функції при першому завантаженні компонента (Лістинг 3.5).

Лістинг 3.6 – Функція підключення гаманця

```
const connectWallet = async () => {
  try {
    if (!window.ethereum)
      return setOpenError(true), setError("Install MetaMask");
    const accounts = await window.ethereum.request({
      method: "eth_requestAccounts",
    });
    setCurrentAccount(accounts[0]);
    connectingWithSmartContract();
  } catch (error) {}
};
```

Функція connectWallet підключає гаманець користувача до додатку. Вона відкриває MetaMask для вибору акаунта та встановлює поточний акаунт у стані. Після успішного підключення вона також викликає функцію connectingWithSmartContract для підключення до смарт-контракту (Лістинг 3.6).

Лістинг 3.7 – Завантаження файлів на IPFS через Pinata

```
const uploadToPinata = async (file) => {
```

```
    if (file) {
      try {
        const formData = new FormData();
        formData.append("file", file);
        const response = await axios({
          method: "post",
          url: "https://api.pinata.cloud/pinning/pinFileToIPFS",
          data: formData,
          headers: {
            pinata_api_key: `ceb33d5565e1e5d62c0`,
            pinata_secret_api_key:
`2241e1d08bba20e8205118443ade50d615a0281514dff856809ee91e30ffde3`,
            "Content-Type": "multipart/form-data",
          },
        });
        const ImgHash =
`https://gateway.pinata.cloud/ipfs/${response.data.IpfsHash}`;
        return ImgHash;
      } catch (error) {
        console.log("Unable to upload image to Pinata");
      }
    }
  };
```

Функція uploadToPinata завантажує файл на IPFS через Pinata. Вона приймає файл, створює форму даних, додає до неї файл, а потім відправляє POST-запит на API Pinata. У разі успіху вона повертає хеш зображення з IPFS (Лістинг 3.7).

Лістинг 3.8 – Створення NFT

```
const createNFT = async (name, price, image, description, router) => {
    if (!name || !description || !price || !image)
      return setError("Data Is Missing"), setOpenError(true);
    const data = JSON.stringify({ name, description, image });
    try {
      const response = await axios({
        method: "POST",
        url: "https://api.pinata.cloud/pinning/pinJSONToIPFS",
```

```
        data: data,
        headers: {
          pinata_api_key: `ceb33d5565e1e5d62c0`,
          pinata_secret_api_key:
`2241e1d08bba20e8205118443ade50d615a0281514dff856809ee91e30ffde3`,
          "Content-Type": "application/json",
        },
      });
      const url =
`https://gateway.pinata.cloud/ipfs/${response.data.IpfsHash}`;
      console.log(url);
      await createSale(url, price);
      router.push("/searchPage");
    } catch (error) {
      setError("Error while creating NFT");
      setOpenError(true);
    }
  };
```

Функція createNFT створює новий NFT, завантажуючи метадані (ім'я, опис, зображення) на IPFS через Pinata, а потім викликає функцію createSale для додавання NFT на маркетплейс. Якщо будь-яке поле відсутнє, функція сповіщує помилку. У разі успіху користувач перенаправляється на сторінку пошуку, де побачить свою NFT (Лістинг 3.8).

Лістинг 3.9 – Виставлення токену на продаж

```
const createSale = async (url, formInputPrice, isReselling, id) => {
  try {
    const price = ethers.utils.parseUnits(formInputPrice, "ether");
    const contract = await connectingWithSmartContract();
    const listingPrice = await contract.getListingPrice();
    const transaction = !isReselling
      ? await contract.createToken(url, price, {
          value: listingPrice.toString(), })
      : await contract.resellToken(id, price, {
          value: listingPrice.toString(), });
    await transaction.wait();
  } catch (error) {
```

```
    setError("error while creating sale");
    setOpenError(true);
  }};
```

Функція createSale виставляє NFT на продаж в маркетплейсі. Вона конвертує ціну в потрібний формат, отримує підключення до смарт-контракту та викликає відповідний метод для створення нового токена або його перепродажу. У разі успіху чекає підтвердження транзакції (Лістинг 3.9).

Лістинг 3.10 – Отримання всіх NFT з маркетплейсу

```
const fetchNFTs = async () => {
  try {
    const web3Modal = new Web3Modal();
    const connection = await web3Modal.connect();
    const provider = new ethers.providers.Web3Provider(connection);
    const contract = fetchContract(provider);
    const data = await contract.fetchMarketItems();
    const items = await Promise.all(
      data.map(async ({ tokenId, seller, owner, price: unformattedPrice
}) => {
        const tokenURI = await contract.tokenURI(tokenId);
        const {
          data: { image, name, description },
        } = await axios.get(tokenURI);
        const price = ethers.utils.formatUnits(
          unformattedPrice.toString(),
          "ether"
        );
        return {
          price, tokenId: tokenId.toNumber(), seller, owner, image,
name, description, tokenURI,
        };
      })
    );
    return items;
  } catch (error) {
    console.log(error);
  }
};
```

```
useEffect(() => {
  fetchNFTs();
}, []);
```

Функція fetchNFTs отримує всі доступні NFT з маркетплейсу. Вона підключається до контракту, отримує дані про токени та повертає масив об’єктів з інформацією про кожен токен. useEffect забезпечує виконання цієї функції при першому завантаженні компонента (Лістинг 3.10).

Лістинг 3.11 – Отримання NFT користувача або виставлених користувачем на продаж NFT

```
const fetchMyNFTsOrListedNFTs = async (type) => {
  try {
    if (currentAccount) {
      const contract = await connectingWithSmartContract();
      const data =
        type == "fetchItemsListed"
          ? await contract.fetchItemsListed()
          : await contract.fetchMyNFTs();
      const items = await Promise.all(
        data.map(async ({ tokenId, seller, owner, price: unformattedPrice
}) => {
          const tokenURI = await contract.tokenURI(tokenId);
          const {
            data: { image, name, description },
          } = await axios.get(tokenURI);
          const price = ethers.utils.formatUnits(
            unformattedPrice.toString(),
            "ether"
          );
          return {
            price, tokenId: tokenId.toNumber(), seller, owner, image,
name, description, tokenURI,
          };
        })
      );
      return items;
    }
```

```
  } catch (error) {}
};
useEffect(() => {
  fetchMyNFTsOrListedNFTs();
}, []);
```

Функція fetchMyNFTsOrListedNFTs залежно від переданого типу отримує або NFT якими користувач володіє, або виставлені ним на продаж NFT. Вона підключається до контракту та повертає масив об'єктів з інформацією про кожен токен (Лістинг 3.11).

Лістинг 3.12 – Функція купівлі NFT

```
const buyNFT = async (nft) => {
  try {
    const contract = await connectingWithSmartContract();
    const price = ethers.utils.parseUnits(nft.price.toString(), "ether");
    const transaction = await contract.createMarketSale(nft.tokenId, {
      value: price,
    });
    await transaction.wait();
    router.push("/author");
  } catch (error) {
    setError("Error While buying NFT");
    setOpenError(true);
  }
};
```

Функція buyNFT дозволяє користувачу купити NFT. Вона підключається до контракту, конвертує ціну в потрібний формат, викликає метод createMarketSale для здійснення продажу та чекає підтвердження транзакції. У разі успіху користувач перенаправляється на сторінку автора (Лістинг 3.12).

3.5 Результати розробки вебсайту для взаємодії зі смарт-контрактом на блокчейні

В результаті було розроблено функціональний вебсайт для взаємодії зі смарт-контрактом на блокчейні, який дозволяє користувачам створювати,

купувати та продавати NFT. У цьому розділі продемонстровано основні можливості та функціональність вебсайту.

При першому вході сайт одразу пропонує підключити криптогаманець (Рисунок 3.6).

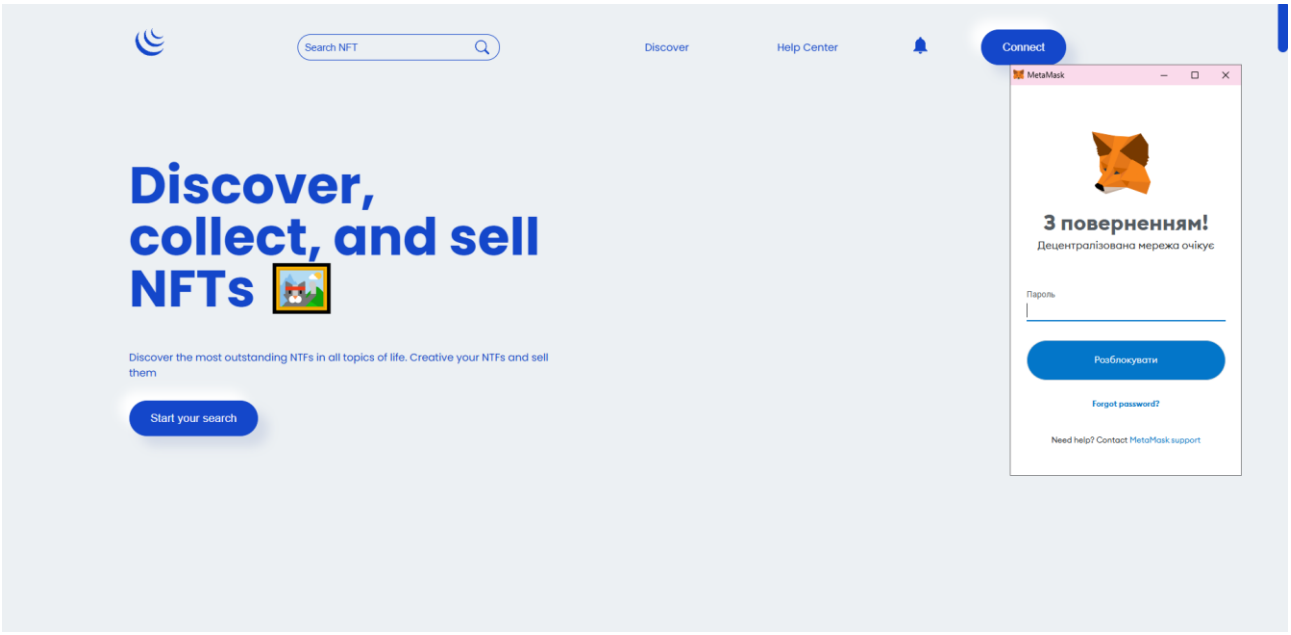

Рисунок 3.6 – Підключення гаманця

Після вводу пароля від гаманця користувач буде авторизований. З’явиться аватарка та буде показано підключений гаманець (Рисунок 3.7).

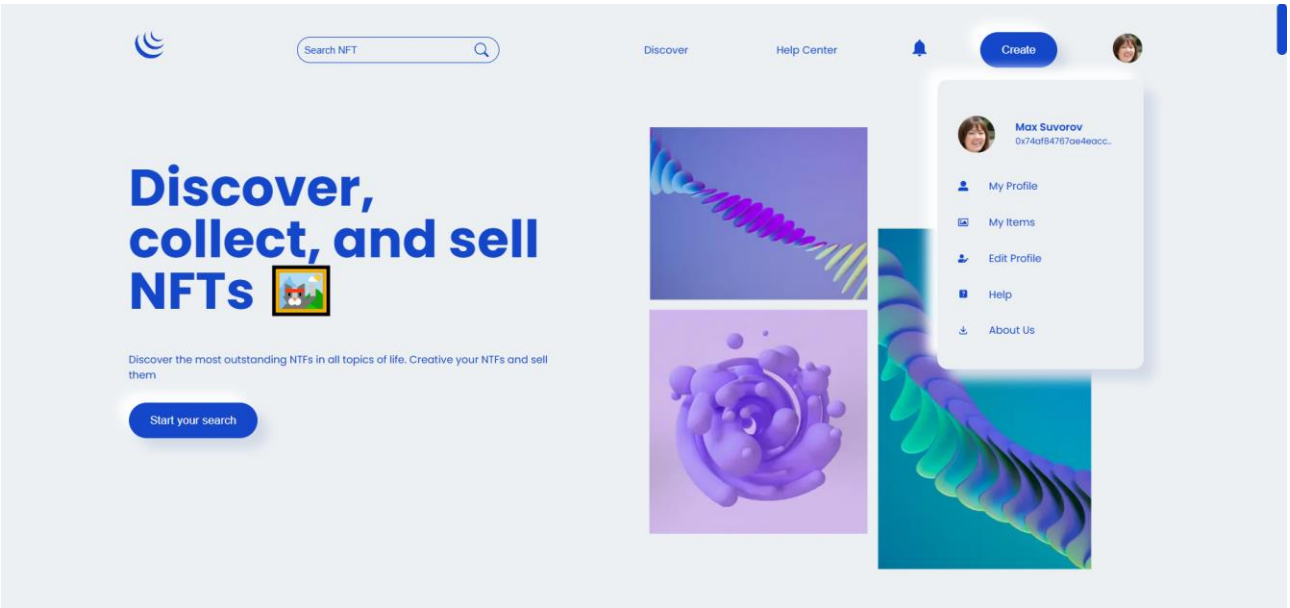

Рисунок 3.7 – Головна сторінка авторизованого користувача

Для створення власної NFT потрібно натиснути на кнопку Create. Користувач буде переадресований на сторінку для заповнення даних (Рисунок 3.8).

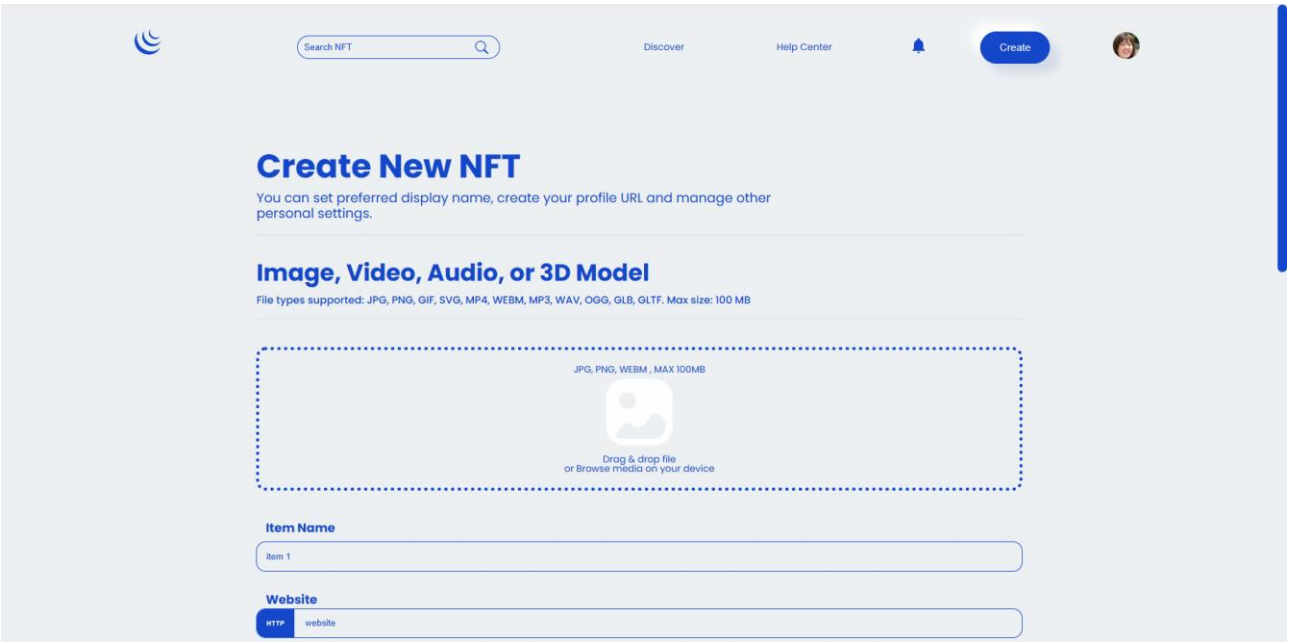

Рисунок 3.8 – Сторінка створення NFT

Спочатку потрібно завантажити файл натиснувши на поле для файлу або перетягнувши файл у відповідну область (Рисунок 3.9).

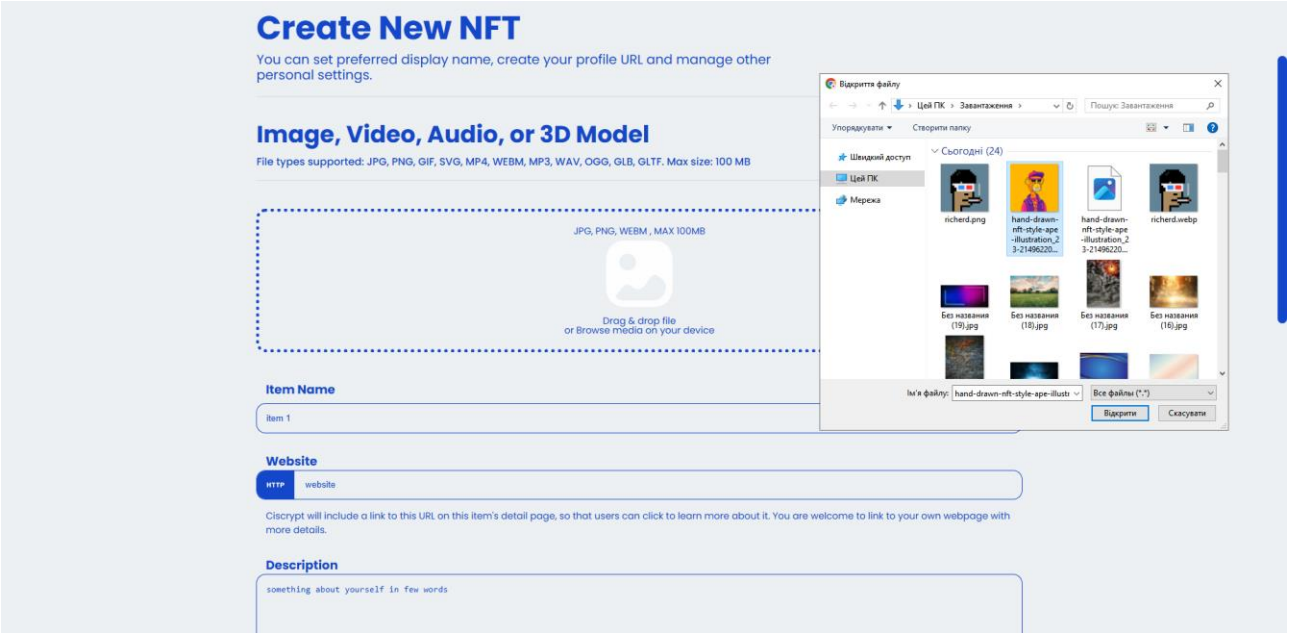

Рисунок 3.9 – Завантаження файлу

Файл одразу з’являється у Pinata (Рисунок 3.10) і відображується на сайті (Рисунок 3.11).

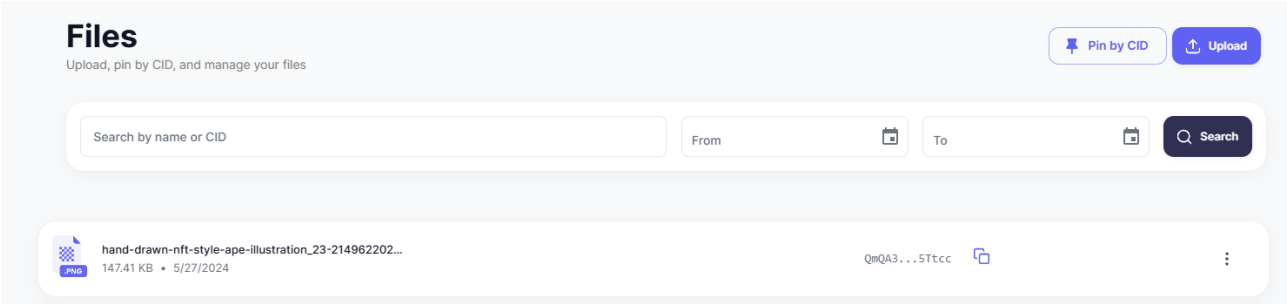

Рисунок 3.9 – Файл у Pinata

Рисунок 3.10 – Відображення файлу на сайті

Після завантаження файлу полрібно заповнити форму, де вказуються назва, посилання на сайт або портфоліо, опис, обирається колекція. Далі вказуються ціна у ETH, властивості та відсоток, який буде передаватись автору з кожного продажу NFT (Рисунок 3.11). При натисканні на кнопку Upload з’явиться вікно Metamask з платним підписом для підтвердження створення NFT (Рисунок 3.12).

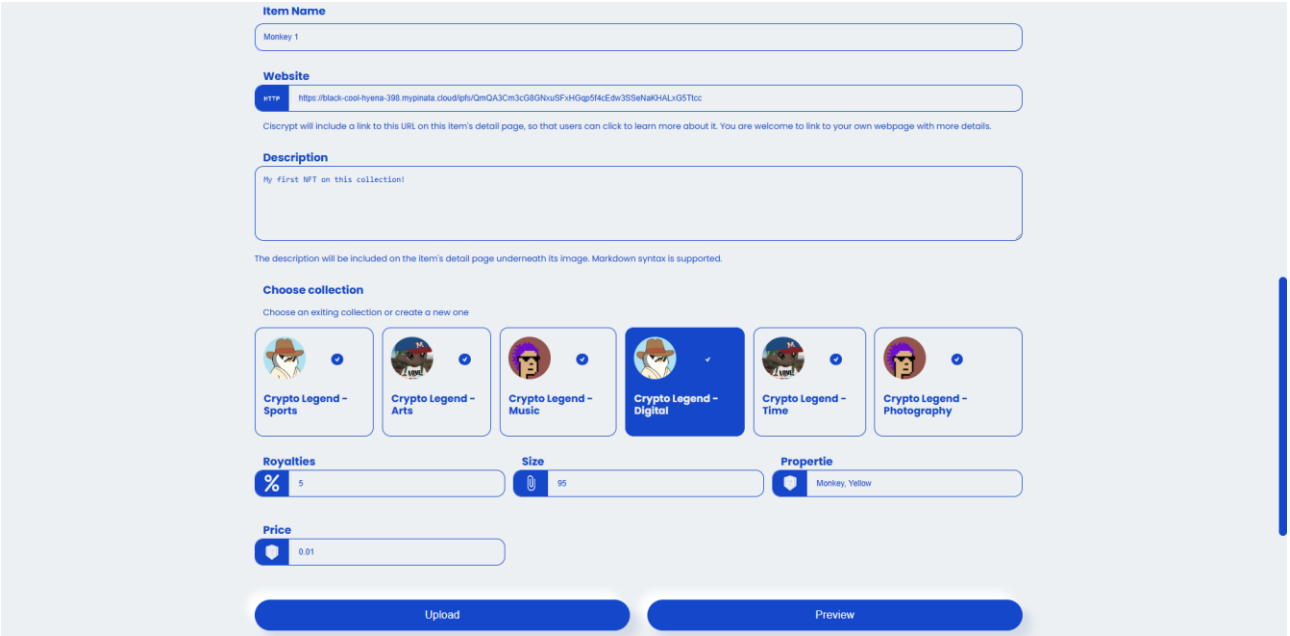

Рисунок 3.11 – Заповнення форми

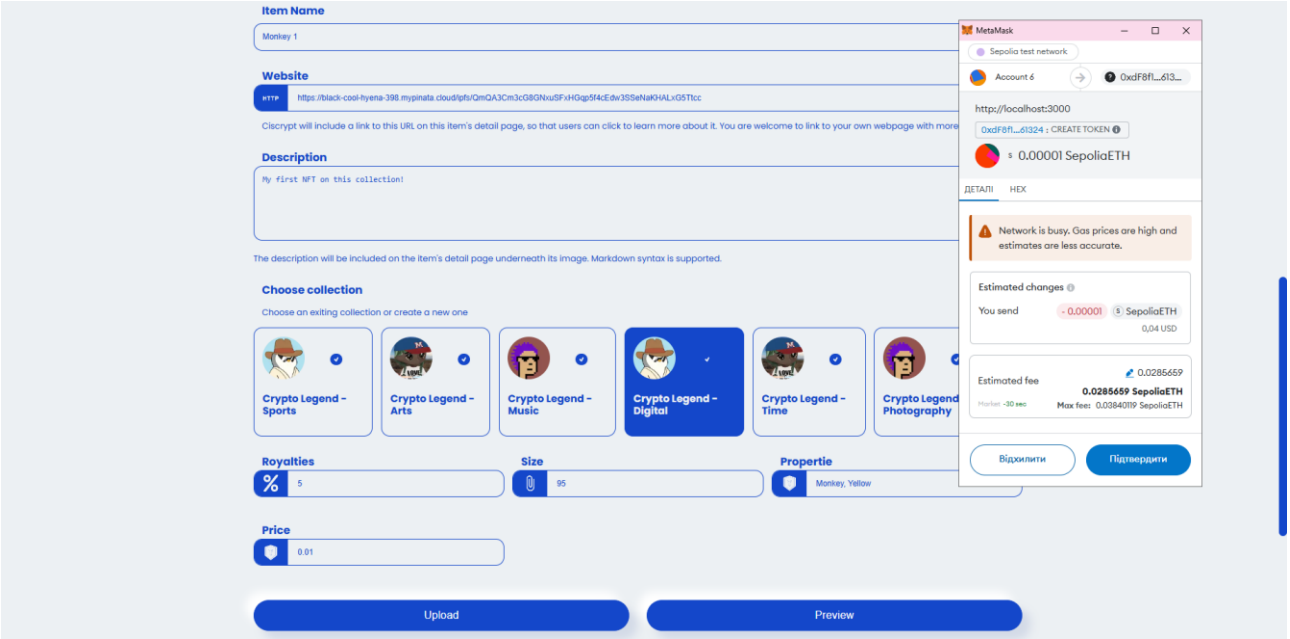

Рисунок 3.12 – Підпис транзакції на створення NFT

Після підписання транзакції, її можна подивитись у блокчейн провіднику Etherscan (Рисунок 3.13). У провіднику можна побачити що тразнакція коштувала приблизно 0.02 ETH, а 0.00001 ETH відправилось на смарт-контракт у вигляді комісії. Також видно що NFT спочатку з’явилась на адресі криптогаманця і одразу відправилась на адресу смарт-контракта, оскільки була автоматично виставлена на продаж.

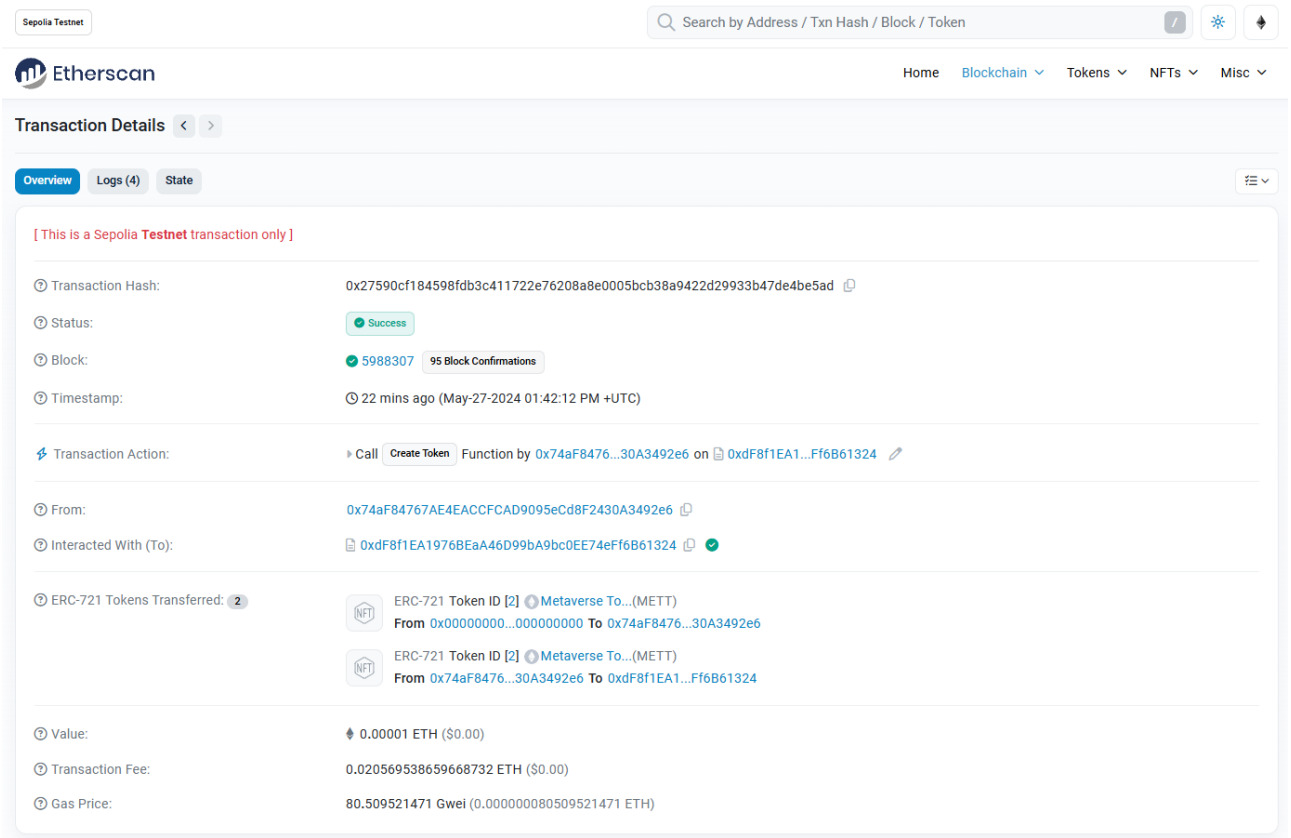

Рисунок 3.13 – Транзакція створення NFT у блокчейн провіднику

У Pinata можна побачити збережені назву NFT, опис та посилання на збережений раніше файл (Рисунок 3.14).

{"name":"Monkey 1","description":"My first NFT on this collection!","image":"https://gateway.pinata.cloud/ipfs/QmQA3Cm3cG8GNxuSFxHGqp5f4cEdw3SSeNaKHALxG5Ttcc"}

Рисунок 3.14 – Дані створеної NFT у Pinata

На маркетплейсі можна відкрити сторінку із створеною NFT. (Рисунок 3.15). На ній відображені ціна за якою виставлена NFT, опис, розмір картинки, адресу смарт-контракта на якій вона зберігається. Оскільки авторизований користувач є автором цієї NFT, в нього немає кнопки для покупки. Якщо авторизуватись на сайті з іншого гаманця, то кнопка буде відображена. Натиснувши на неї відкриється вікно Metamask з платним підписом для підтвердження купівлі NFT (Рисунок 3.16).

Рисунок 3.15 – Створена NFT

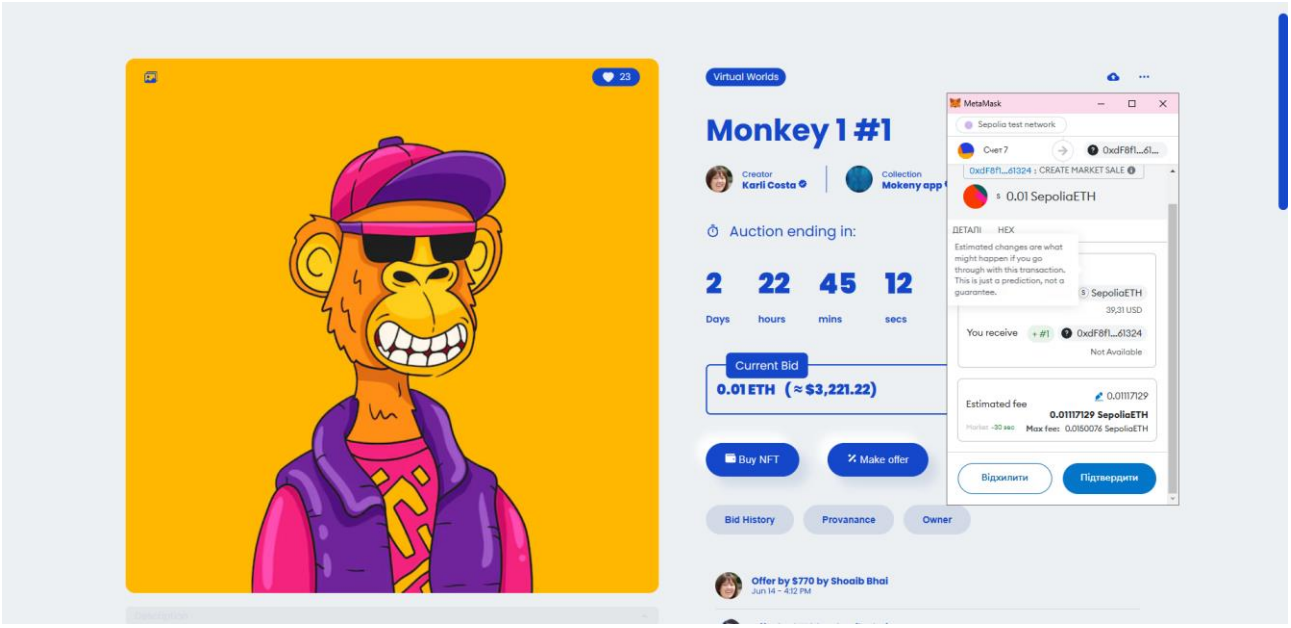

Рисунок 3.16 – Купівля NFT з другого гаманця

Знову дивимось транзакцію у блокчейн провіднику Etherscan і бачимо що комісія мережі за тразнакцію коштувала приблизно 0.01 ETH, а 0.00001 ETH відправилось на смарт-контракт у вигляді комісії. Також відправилось 0.01 ETH на смарт-контракт, а контракт переслав їх на криптогаманець продавця, стільки коштувала NFT. Після цього смарт-контракт відправив NFT покупцю (Рисунок 3.17).

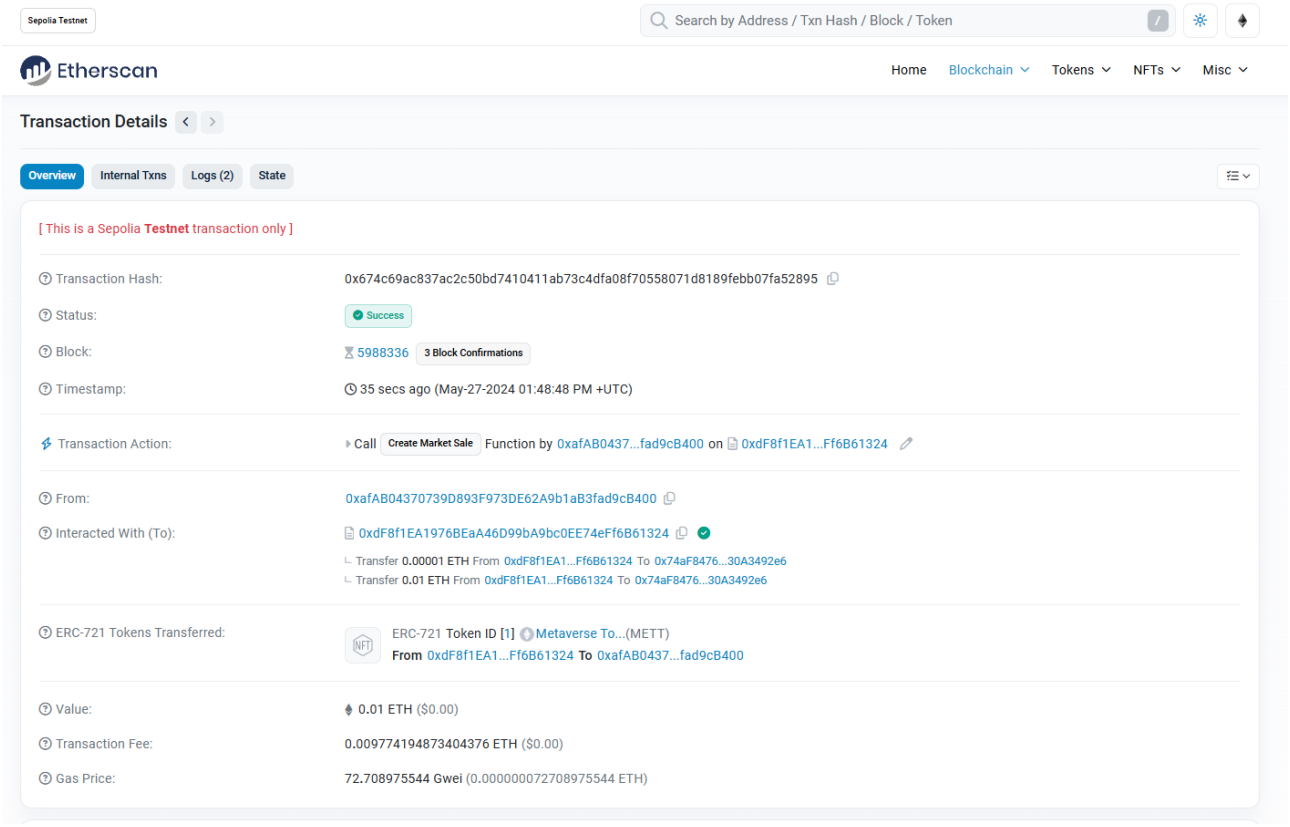

Рисунок 3.17 – Транзакція купівлі NFT у блокчейн провіднику

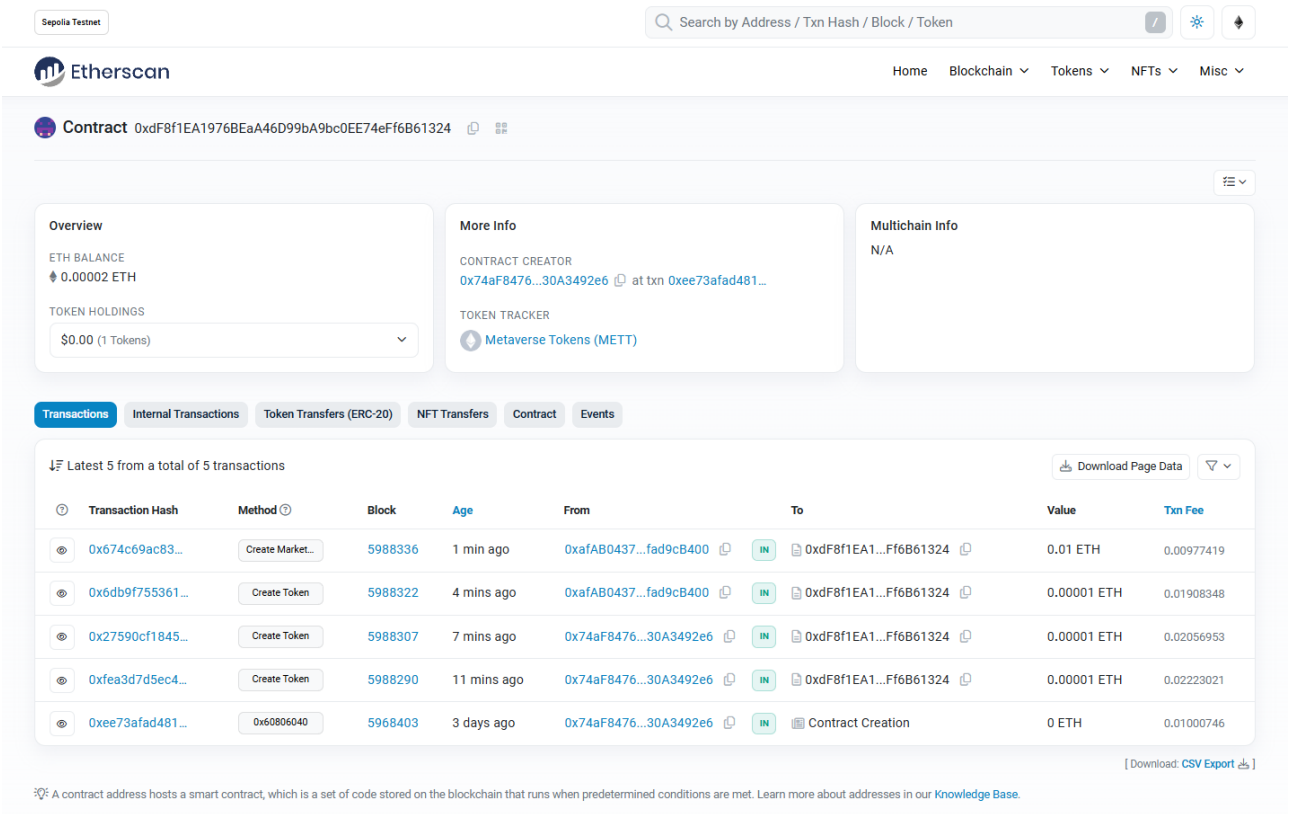

Рисунок 3.18 – Сторінка смарт-контракту у блокчейн провіднику

Якщо відкрити сторінку смарт-контракту у блокчейн провіднику, то можна подивитись усі минулі транзакції, переміщення криптовалют та баланс контракту (Рисунок 3.18). На балансі контракту 0.00002 ETH, які прийшли з комісій за створення та продаж нфт, та 1 токен (NFT), який було створено з іншого гаманця.

На сторінці для перегляду NFT можна ознайомитись зі своїми NFT, розділеними на дві категорії (Рисунок 3.19). Перша категорія це NFT які виставлені на продаж (Listed NFTs), а друга – це усі NFT, що знаходяться на криптогаманці (Own NFT).

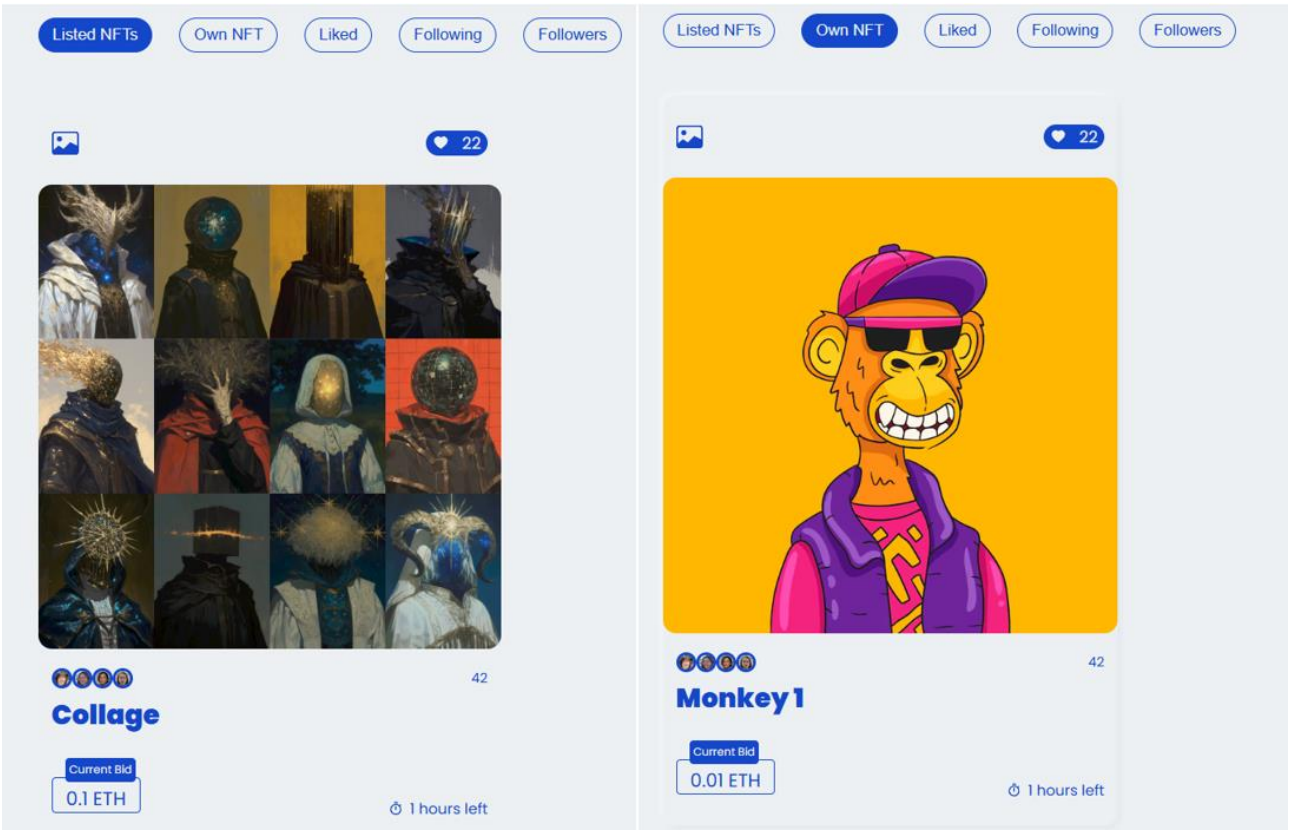

Рисунок 3.19 – Listed та Own NFT

ВИСНОВКИ

Проведене дослідження дозволило проаналізувати вплив технології блокчейн та NFT на розвиток автоматизованих систем у мистецтві. Було вивчено сучасний стан використання цих технологій, розглянуто існуючі методи їх реалізації, а також визначено їхні можливості та перспективи у цій галузі.

Технологія блокчейн має значний потенціал для трансформації арт-індустрії. Її ключові характеристики, такі як децентралізація, прозорість, безпека та незмінність даних, можуть вирішити низку проблем, з якими стикається мистецька сфера, включаючи підробки творів мистецтва, порушення авторських прав, відсутність прозорості на арт-ринку та складнощі з залученням інвестицій.

NFT відкривають нові можливості для створення, продажу та колекціонування цифрового мистецтва. Ці унікальні цифрові активи дозволяють художникам отримувати винагороду за свою роботу без посередників, а колекціонерам – безпечно купувати та володіти цифровими творами мистецтва.

Існуючі методи реалізації технологій блокчейн та NFT у мистецтві мають певні недоліки. Серед них – високі комісії за транзакції, складність використання, ризик спекуляцій, проблеми масштабованості, правова невизначеність та питання екологічного впливу.

Необхідно розробляти нові рішення та підходи, які б усували недоліки існуючих методів. Зокрема, важливо створювати децентралізовані платформи, що підтримують роботу з різними блокчейнами, мають універсальний характер, пропонують різні формати аукціонів, використовують методи оптимізації смарт-контрактів для зниження комісій та мають зручний інтерфейс для користувачів.

У практичній частині роботи було висвітлено проектування та розробку NFT-маркетплейса. В результаті розроблено смарт-контракти маркетплейсу та англійського і голландського аукціонів. Оптимізовано за допомогою Yul контракт маркетплейсу. Спроектовано діаграму прецедентів NFT-маркетплейсу та діаграми послідовностей дій купівлі, продажу та виставлення на продаж NFT. Розроблено вебсайт для взаємодії зі смарт-контрактом на блокчейні.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Суворов М.В. Дослідження технології блокчейн для проектування та розробки автоматизованих систем у мистецтві: матеріали 28-го Міжнар. молодіж. форуму, 2024. 640-641 с.
2. Bitcoin: A Peer-to-Peer Electronic Cash System : електронний документ. URL: https://bitcoin.org/bitcoin.pdf (дата звернення: 03.01.2024).
3. Dependable IoT using Blockchain-based Technology : електронний документ. URL: https://repositorio.pucrs.br/dspace/bitstream/10923/22146/2/Dependable_IoT_using_Blockchainbased_Technology.pdf (дата звернення: 07.02.2024).
4. Ethereum : вебсайт. URL: https://ethereum.org/en/ (дата звернення: 04.01.2024).
5. Layer 2 : вебсайт. URL: https://academy.binance.com/en/glossary/layer-2 (дата звернення: 16.02.2024).
6. What Is Proof of Work (PoW) in Blockchain? : вебсайт. URL: https://www.investopedia.com/terms/p/proof-work.asp (дата звернення: 18.02.2024).
7. What is proof of stake? : вебсайт. URL: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-proof-of-stake (дата звернення: 18.02.2024).
8. Types of Blockchains Explained: Public vs Private vs Consortium : вебсайт. URL: https://www.blockchain-council.org/blockchain/types-of-blockchains-explained-public-vs-private-vs-consortium/ (дата звернення: 24.02.2024).
9. Non Fungible Token (NFT) Standards: An Overview : вебсайт. URL: https://hackernoon.com/non-fungible-token-nft-standards-an-overview-w71y34y3 (дата звернення: 01.02.2024).
10. A Complete Guide to NFTs – Definition, Minting, and Tech Specs : вебсайт. URL: https://web3.hashnode.com/a-complete-guide-to-nfts-definition-minting-and-tech-specs (дата звернення: 03.02.2024).
11. EIP-721 : вебсайт. URL: https://eips.ethereum.org/EIPS/eip-721 (дата звернення: 26.02.2024).
12. EIP-1155 : вебсайт. URL: https://eips.ethereum.org/EIPS/eip-1155 (дата звернення: 26.02.2024).

13. NFT videos: Everything you need to know in 2023 : вебсайт. URL: https://storyxpress.co/blog/nft-videos/ (дата звернення: 05.02.2024).

14. Kalyta N.I., Lymar L.V. Research of the efficiency of using blockchain technology in gaming applications. Sworldjournal. 2023. Vol. 1. P. 3–10.

15. What Are Utility NFTs? : вебсайт. URL: https://builtin.com/nft-non-fungible-token/utility-nft (дата звернення: 29.02.2024).

16. A Beginner's Guide on the Legal Risks and Issues Around NFTs : вебсайт. URL: https://cointelegraph.com/learn/a-beginners-guide-on-the-legal-risks-and-issues-around-nfts (дата звернення: 15.02.2024).

17. The Non-Fungible Token Bible: Everything you need to know about NFTs : вебсайт. URL: https://blog-v3.opensea.io/articles/non-fungible-tokens (дата звернення: 10.02.2024).

18. Antonopoulos, A. M., Wood, G. Mastering Ethereum: Building Smart Contracts and DApps. Sebastopol : O'Reilly Media, 2018. 424 p.

19. Dannen, C. Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners. Berkeley: Apress, 2017. 255 p.

20. Documentation: Yul : вебсайт. URL: https://docs.soliditylang.org/en/latest/yul.html (дата звернення: 15.03.2024).

21. Krishnan V. The Essential Guide to Web3. 1st edition. Birmingham : Packt Publishing, 2023. 366 p.

22. Tiwari, V., Singh, S. K., Vadi, V. R. Smart Contract Using Solidity (Remix -Ethereum IDE). International Journal of Advanced Research in Computer and Communication Engineering. 2023. Vol. 12. No. 2. P. 243–249.

23. Banks, A., Porcello, E. Learning React: Modern Patterns for Developing React Apps. 2nd ed. Sebastopol : O'Reilly Media, 2020. 350 p.

24. Wieruch, R. The Road to Learn React: Your Journey to Master Plain Yet Pragmatic React.js. Berlin : Leanpub, 2020. 200 p.

25. Duckett J. HTML and CSS: Design and Build Websites. 1st edition. Indianapolis: Wiley, 2011. 512 p.

26. Riva M. Real-World Next.js: Build scalable, high-performance, and modern web applications using Next.js, the React framework for production. Birmingham: Packt Publishing, 2022. 348 p.

27. Tazetdinov A. Next.js Cookbook. London: BPB Online, 2023. 270 p.

28. Jeffery Owens. Ethers js: The Ultimate Guide to Building and Deploying Ethereum Applications. Independently published, 2023. 167 p.

29. Shashank M. A Brief Introduction to Web3: Decentralized Web Fundamentals for App Development. 1st edition. New York: Apress, 2022. 199 p.

30. Kolce J., Brown M., Buckler C., Wanyoike M., Jacques N. Modern JavaScript Tools & Skills. Melbourne: SitePoint, 2018. 47 p.

31. CryptoGuy V. A Beginner's Guide to METAMASK: The Most Powerful Hot Wallet in the Crypto Space. Independently published, 2022. 53 p.

32. Raj K. Foundations of Blockchain. Mumbai: Packt Publishing, 2019. 372 p.

33. Cockburn A. Writing Effective Use Cases. Addison-Wesley Professional, 2000. 304 p.

34. Ambler S. The Elements of UML™ 2.0 Style. Cambridge University Press, 2005. 202 p.