Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти перший (бакалаврський)

Програмні компоненти для токенізації освітних активів на основі смарт-контрактів Ethereum

(тема)

Виконав:
здобувач 4 року навчання,
групи КІУКІ-21-5
Олексій ЛІЗУНОВ
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник: доц. Олександр ШМАТКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Лізунову Олексію В'ячеславовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Програмні компоненти для токенізації освітних активів на основі смарт-Контрактів Ethereum

затверджена наказом по університету від “ 26 ” травня 2025 р. № 424 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 17 червня 2025 р.

3. Вхідні дані до роботи
Платформа Ethereum
Мережа блокчейн

4. Перелік питань, що потрібно опрацювати у роботі
1) Огляд літератури за темою роботи;
2) Аналіз предметної області;
3) Вибір та обґрунтування методики дослідження;
4) Проведення експериментальних досліджень;
5) Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Слайд-презинтація – 10 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Аналіз проблеми та огляд існуючих рішень	27.05.25-30.05.25	
2	Вибір технології розробки та інструменталь-	31.05.25-02.06.25	
	них засобів		
3	Розробка алгоритмічного забезпечення	03.06.25-05.06.25	
4	Розробка та відлагодження програмного	06.06.25-09.06.25	
	забезпечення		
5	Оформлення матеріалів кваліфікаційної роботи	10.06.25-11.06.25	
6	Подання кваліфікаційної роботи керівникові та	12.06.25-13.06.25	
	її попередній захист		
7	Подання кваліфікаційної роботи на рецензу-	14.06.25-16.06.25	
	вання		

Дата видачі завдання " 26 " травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Олександр ШМАТКО
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 94 с., 24 рис., 5 табл., 1 дод., 18 джерел.

ETHEREUM, СМАРТ-КОНТРАКТ, БЛОКЧЕЙН, ДЕЦЕНТРАЛІЗАЦІЯ, АЛГОРИТМ КОНСЕНСУСУ, НОДА, ТОКЕНІЗАЦІЯ, STAKING, SHA-256, ECDSA, WEB3, IPFS, L2-РІШЕННЯ, ENS, DAPP, METAMASK.

Метою кваліфікаційної роботи є дослідження, проєктування та розробка програмних компонентів для токенізації освітніх активів з використанням блокчейн-технологій на платформі Ethereum.

У рамках роботи було створено децентралізований вебзастосунок (dApp), який дозволяє користувачам взаємодіяти зі смарт-контрактом Ethereum для збереження та перегляду освітніх повідомлень. Для реалізації токенізації та зберігання даних запропоновано використання технологій блокчейн, смарт-контрактів, а також інструментів фронтенд-розробки, зокрема React.js та Web3.js. Розроблений додаток включає адаптивний інтерфейс, підтримку перемикання теми, інтеграцію з MetaMask, обробку облікових записів і балансів, а також запис/читання повідомлень у блокчейні.

У результаті проведеної роботи був створений прототип системи, яка демонструє можливості токенізації освітніх активів. У подальшому розроблений прототип може бути суттєво масштабований та розширений до повноцінної платформи, що дозволяє зберігати, автентифікувати та перевіряти сертифікати, дипломи, освітні курси, навички та досягнення студентів та випускників вищих навчальних закладів.

ABSTRACT

Bachelor’s thesis: 94 pages, 24 figures, 5 tables, 1 appendices, 18 sources.

ETHEREUM, SMART CONTRACT, BLOCKCHAIN, DECENTRALIZATION, CONSENSUS ALGORITHM, NODE, TOKENIZATION, STAKING, SHA-256, ECDSA, WEB3, IPFS, L2 SOLUTIONS, ENS, DAPP, METAMASK.

The goal of this qualification work is to research, design, and develop software components for the tokenization of educational assets using blockchain technologies on the Ethereum platform.

As part of the work, a decentralized web application (dApp) was created, allowing users to interact with an Ethereum smart contract for storing and viewing educational messages. To implement tokenization and data storage, the use of blockchain technologies, smart contracts, and frontend development tools–particularly React.js and Web3.js–was proposed. The developed application includes a responsive interface, theme switching support, MetaMask integration, account and balance handling, as well as reading/writing messages on the blockchain.

As a result of the project, a prototype system was developed to demonstrate the possibilities of educational asset tokenization. This prototype can be significantly scaled and expanded in the future into a fully-fledged platform that enables the storage, authentication, and verification of certificates, diplomas, educational courses, skills, and achievements of students and graduates of higher education institutions.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

BTC (Bitcoin) – криптовалюта Біткоїн

COMP (Compound) – протокол децентралізованого кредитування Compound

DAG (Directed Acyclic Graph) – орієнтований ациклічний граф

DAO (Decentralized Autonomous Organization) – децентралізована автономна організація

DAPP (Decentralized Application) – децентралізований додаток

DPoS (Delegated Proof of Stake) – делегований доказ володіння часткою

ECDSA (Elliptic Curve Digital Signature Algorithm) – алгоритм цифрового підпису на основі еліптичних кривих

ENS (Ethereum Name Service) – система імен Ефіріум

ETC (Ethereum Classic) – блокчейн-платформа Ефіріум Класік

ETH (Ethereum) – блокчейн-платформа Ефіріум

IPFS (InterPlanetary File System) – міжпланетна файлова система

L2 (Layer 2) – рішення другого рівня

LDO (Lido DAO) – децентралізована автономна організація Lido

LS (Liquid Staking) – ліквідний стейкінг

OP (Optimistic Rollup) – метод масштабування блокчейну Optimistic Rollup

PKI (Public Key Infrastructure) – інфраструктура відкритих ключів

PoId (Proof of Identity) – доказ ідентичності

PoS (Proof of Stake) – доказ володіння часткою

PoW (Proof of Work) – доказ виконання роботи

SHA-256 (Secure Hash Algorithm) – криптографічна хеш-функція

ZKSYNC (Zero-Knowledge Rollups) – метод масштабування блокчейну з використанням доказів із нульовим розголошенням

ВСТУП

Сьогодні ми спостерігаємо, як світ активно переходить у цифровий формат, змінюючи фундаментальні аспекти життя суспільства – економіку, освіту, медицину та державне управління. Цифрова трансформація передбачає не лише автоматизацію процесів, а й фундаментальну зміну підходів до організації діяльності за рахунок використання цифрових платформ, децентралізованих рішень та інтелектуальних технологій.

Одним із провідних напрямів цієї трансформації є розвиток наскрізних цифрових технологій, серед яких особливе місце займає система розподіленого реєстру (блокчейн). Разом із великими даними, штучним інтелектом, квантовими обчисленнями, віртуальною реальністю та бездротовими сенсорними мережами, блокчейн формує технічну основу нового цифрового суспільства.

Застосування технології блокчейн у сфері освіти відкриває нові можливості для верифікації досягнень, цифрової ідентифікації, захисту інтелектуальної власності, а також для створення доказової бази освітніх результатів у вигляді токенізованих активів. У контексті розвитку децентралізованих платформ, що базуються на Ethereum, можливість токенізації освітніх даних набуває особливої актуальності.

Метою кваліфікаційної роботи є дослідження, проєктування та розробка програмного компонента для токенізації освітніх активів на основі блокчейн-платформи Ethereum із використанням смарт-контрактів, технологій Web3 та клієнтської частини на базі React.js.

У межах поставленої мети необхідно вирішити такі завдання:

- провести аналіз методів забезпечення конфіденційності та цілісності даних у блокчейн-системах;
- дослідити принципи функціонування хеш-алгоритмів у контексті захисту даних;

- розглянути інструменти збереження приватності в децентралізованих реєстрах;

- розробити прототип децентралізованого застосунку для токенізації освітніх активів на основі Ethereum;

- провести тестування системи та оцінити її придатність до практичного впровадження.

Наукова новизна дослідження полягає у:

- системному аналізі тенденцій використання блокчейн-технологій в освітній сфері;

- запропонованій архітектурі програмного рішення для токенізації освітніх активів;

- розробці децентралізованого вебзастосунку на основі Ethereum-смарт-контракту з підтримкою взаємодії через Web3 і MetaMask.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Аналіз поколінь блокчейн-технологій та їх застосування в освіті

Розглядаючи розвиток блокчейн-технологій, варто почати з Blockchain 1.0. Це перше покоління блокчейну, основною сферою застосування якого стали криптовалюти. Його головною метою було створення безпечного, прозорого та децентралізованого механізму передачі цифрових цінностей без необхідності залучення третьої довіреної сторони.

Найяскравішим прикладом цього покоління є Bitcoin, який уперше використав блокчейн як публічну транзакційну книгу. До основних функцій першого покоління відносять забезпечення автентичності даних, перевірку транзакцій, захист від подвійних витрат, а також прозорість і децентралізацію. Технологічно Blockchain 1.0 характеризується використанням механізму консенсусу Proof of Work (PoW).

Blockchain 2.0 суттєво розширило сферу застосування блокчейн-технологій завдяки впровадженню смарт-контрактів та децентралізованих застосунків (DApps). Смарт-контракти представляють собою програмовані угоди, які автоматично виконуються після досягнення заздалегідь заданих умов. DApps – це децентралізовані додатки, що працюють на основі таких смарт-контрактів, забезпечуючи автоматизацію та прозорість різноманітних процесів. Прикладом платформ другого покоління є Ethereum, який використовує не тільки PoW, але й інші механізми консенсусу, такі як Proof of Stake (PoS) чи Proof of Authority (PoA).

Blockchain 3.0 фокусується на подоланні обмежень попередніх поколінь, забезпечуючи високий рівень масштабованості, можливість інтеграції з іншими системами, а також покращену гнучкість і економічність. Для масштабування транзакцій використовуються технології другого рівня (Layer 2), такі як zk-Rollups та Optimistic Rollups, що дозволяють значно

підвищити швидкість обробки транзакцій. Важливу роль також відіграє інтероперабельність – можливість ефективного обміну даними між різними блокчейн-системами. Приклади технологій третього покоління – Polkadot, Cosmos та Cardano, які підтримують міжланцюгові функції та більш досконалі моделі консенсусу.

Blockchain 4.0 є наступним етапом еволюції технології, який орієнтований на інтеграцію блокчейну в реальні галузеві процеси – від бізнесу до освіти, охорони здоров'я і державного управління. Його головною метою є створення індустріально адаптованих рішень, які комбінують блокчейн з іншими високотехнологічними напрямами, такими як штучний інтелект (AI), великі дані (Big Data) та Інтернет речей (IoT).

Приклади таких інтеграцій включають платформу Hyperledger, EOSIO та Dragonchain, які пропонують інтелектуальні смарт-контракти, безпечну передачу даних між пристроями, підтримку приватності та безпеки з використанням zk-SNARKs, гомоморфного шифрування та децентралізованої ідентифікації (DID). Також важливою особливістю четвертого покоління є модульна архітектура, що дозволяє швидко інтегрувати блокчейн у вже існуючі системи завдяки зручним API та SDK. У сфері освіти блокчейн четвертого покоління дозволяє ефективно верифікувати дипломи, вести облік досягнень і створювати надійну цифрову ідентифікацію студентів.

1.2 Потенціал блокчейн-технологій для цифрової трансформації освіти

Еволюція блокчейн-технологій від першого до четвертого покоління демонструє перехід від базових криптовалют до повноцінних децентралізованих екосистем із розширеним функціоналом, орієнтованих на реальні сценарії застосування, зокрема в освітній сфері.

Блокчейн має потенціал радикально трансформувати освітні процеси, підвищуючи прозорість, довіру, безпеку та ефективність обміну інформацією між учасниками освітньої екосистеми: студентами, навчальними закладами,

роботодавцями, державними установами.

Однією з важливих переваг блокчейн-технологій є підвищення автономії студентів. Завдяки можливості зберігати свої навчальні досягнення, сертифікати та навички у вигляді токенів, прив’язаних до персонального криптовалютного гаманця, студенти отримують повний контроль над своєю цифровою ідентичністю. Це дозволяє створювати власне цифрове портфоліо, яке студенти можуть самостійно пред'являти потенційним роботодавцям без посередників.

Також важливою перевагою є підтримка персоналізованого навчання. Блокчейн дозволяє зберігати докладну історію навчального процесу студента, що забезпечує викладачам і освітнім платформам необхідні прозорі та достовірні дані для формування індивідуальних освітніх траєкторій.

Застосування блокчейну забезпечує високий рівень автентичності та конфіденційності освітніх даних. Дипломи, сертифікати та оцінки, зафіксовані в блокчейні, неможливо підробити або змінити заднім числом. Конфіденційна інформація захищається за допомогою криптографічних хеш-функцій, шифрування та технологій Zero Knowledge Proofs. При цьому в блокчейні зберігаються не самі документи, а лише їхні криптографічні відбитки, що забезпечує надійну перевірку їх автентичності без розкриття конфіденційних деталей.

Прозорість та довіра є фундаментальними характеристиками блокчейн-систем. Завдяки незмінності та відкритості інформації у публічному реєстрі, усі учасники освітнього процесу можуть легко і швидко перевіряти автентичність документів, що особливо актуально при працевлаштуванні або подальшому навчанні.

Крім того, блокчейн значно знижує адміністративне навантаження, автоматизуючи процеси видачі, перевірки та передачі дипломів. Це дозволяє відмовитись від паперового документообігу та посередників, пришвидшуючи взаємодію між освітніми установами, роботодавцями та державними органами.

Важливою перевагою є також захист освітніх даних у кризових ситуаціях. Цифрові документи, які зберігаються в блокчейні, є незалежними від локального зберігання. У випадку катастроф, війни чи стихійних лих, важливі освітні документи можна легко відновити з розподіленої мережі, що гарантує їх доступність та збереження.

Завдяки властивості незмінності даних у блокчейні, записані відомості (сертифікати, оцінки, дипломи) є захищеними від фальсифікації. Наприклад, студент не може змінити або видалити свої попередні академічні досягнення, оскільки всі транзакції мають хронологічну послідовність та криптографічний підпис. Це робить блокчейн більш надійним інструментом, ніж паперові документи, які легко підробити або втратити.

Для забезпечення конфіденційності в освітньому середовищі блокчейн не зберігає відкриті особисті дані. Замість цього зберігаються хеші (криптографічні відбитки) від інформації, що дозволяє перевірити достовірність документів без розкриття вмісту. У разі потреби, самі дані можуть бути попередньо зашифровані перед внесенням у блокчейн, або зберігатися в системах на кшталт IPFS, а блокчейн фіксуватиме тільки посилання та хеш-коди.

Блокчейн таким чином виконує роль “якоря довіри” верифікації облікових даних. Це створює передумови для прозорої перевірки сертифікатів роботодавцями та освітніми закладами без потреби у тривалих адміністративних процедурах. Надійність підтверджених даних сприяє підвищенню довіри між учасниками освітнього ринку та ефективнішому поєднанню потреб роботодавців із навичками кандидатів.

Окрім захисту, блокчейн здатен оптимізувати адміністрування освітнього документообігу: автоматизувати видачу дипломів, зменшити залежність від сертифікаційних центрів, пришвидшити перевірку даних. Це особливо важливо у випадках стихійних лих, воєн або катастроф, коли паперові архіви можуть бути знищені. Освітні активи, зафіксовані в децентралізованій системі, залишаються доступними незалежно від фізичної

локації чи стану локальної інфраструктури (рисунок 1.1).

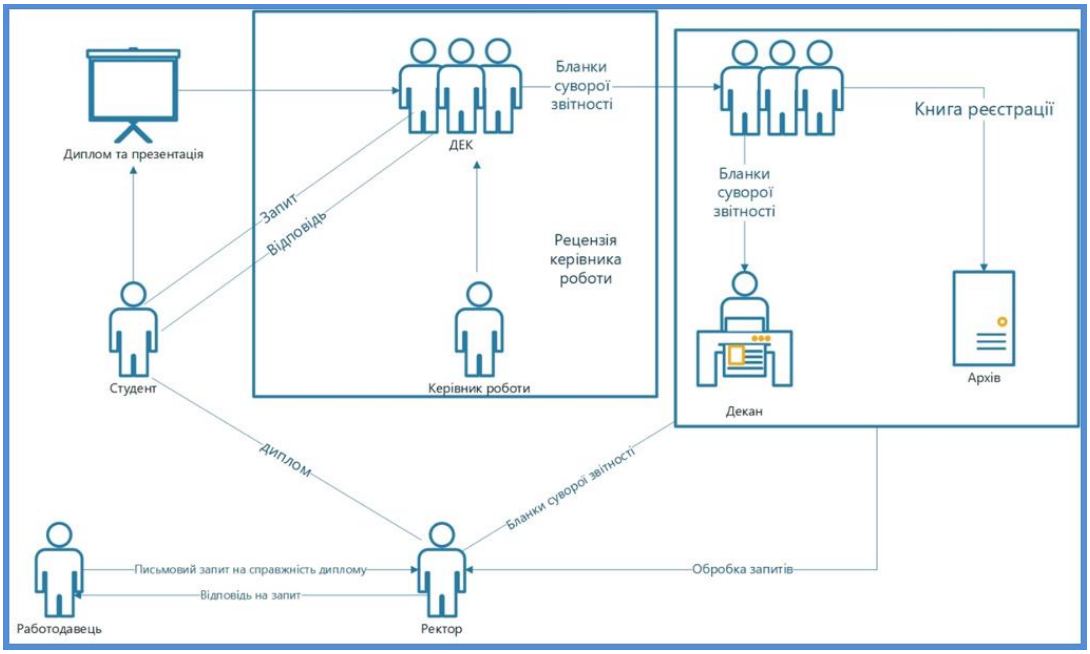

Рисунок 1.1 – Зберігання дипломів та порядок видачі

Таким чином, блокчейн створює надійне середовище для формування цифрової освітньої ідентичності, підвищує безпеку даних та дозволяє автоматизувати критично важливі процеси у сучасній системі освіти.

1.3 Аналіз та дослідження предметної області

Однією з актуальних проблем сучасної освітньої системи є зловживання повноваженнями, бюрократична непрозорість та широке розповсюдження підроблених дипломів, сертифікатів та інших офіційних документів. Такі явища підривають довіру до навчальних закладів, знижують якість освіти та ускладнюють взаємодію з роботодавцями.

Одним із ефективних технологічних рішень цієї проблеми є токенізація освітніх документів з використанням блокчейн-технологій. Такий підхід дозволяє створити незмінну, прозору та децентралізовану базу даних, у якій кожен сертифікат або диплом представлений у вигляді унікального цифрового токена.

Ключові переваги токенізації освітніх активів:

- неможливість фальсифікації чи дублювання сертифікатів;
- пряма верифікація автентичності документа будь-яким користувачем через публічний блокчейн;
- автоматизація перевірки документів без участі третіх сторін;
- швидке виявлення підроблених записів;
- незалежність від фізичного зберігання документів або локальних баз навчальних закладів.

Механізм роботи технології токенізації освітніх активів та запису їх у мережу блокчейн можна розглянути на рисунку 1.2.

Рисунок 1.2 – Принцип роботи технології блокчейн

Схема наочно демонструє процес створення, підпису та збереження цифрового диплома у мережі блокчейн.

На першому етапі вищий навчальний заклад формує токен, який містить основну інформацію: ім’я випускника, назву університету, дату видачі, серійний номер документа тощо.

Далі вміст токена підписується приватним ключем, доступ до якого має лише авторизована освітня установа. Цей цифровий підпис перевіряється мережею вузлів і, після підтвердження, додається до ланцюга блоків у блокчейні.

Таким чином, токенізація документів освіти не лише підвищує захист даних, а й створює новий стандарт довіри між університетами, випускниками та роботодавцями. Принцип дії можна побачити на рисунку 1.3.

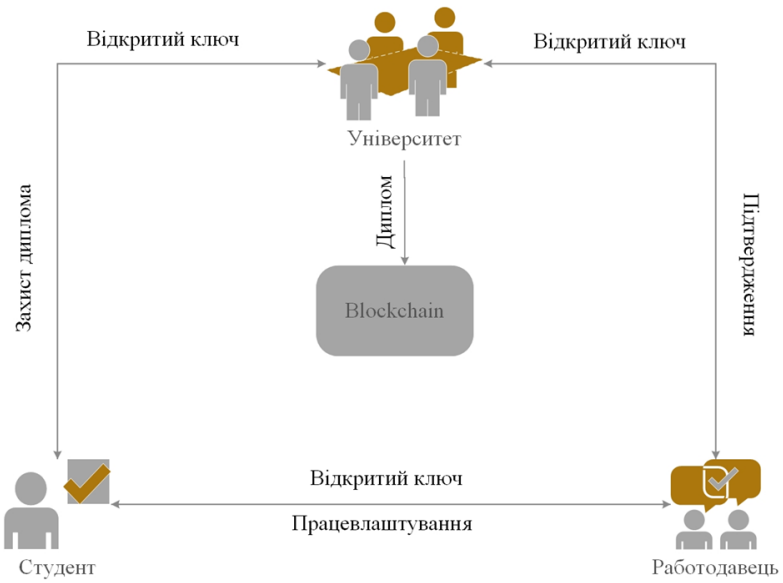

Рисунок 1.3 – Токенізація освітніх активів

Крім цього, зміст диплома хешується – створюється унікальний криптографічний хеш, який забезпечує незмінність та можливість перевірки даних. Якщо в майбутньому буде спроба змінити хоча б один символ у записі, обчислений хеш вже не співпаде з оригінальним. Це дозволяє гарантувати достовірність документа на будь-якому етапі перевірки.

У результаті формування нового блоку створюється запис у реєстрі, а студент отримує відкритий ключ, який може використовуватись для публічного підтвердження чинності диплома.

Такий підхід дозволяє зберігати документи надійно, без втрат, і забезпечує їх верифікацію незалежно від діяльності навчального закладу.

1.4 Аналіз основ для впровадження блокчейн-технології для токенізації

1.4.1 Блокчейн як особиста картка студента

На сьогодні існує велика кількість цифрових платформ, включаючи соціальні мережі, електронні щоденники та освітні сервіси, що дозволяють фіксувати досягнення користувачів. Проте жодна з них не забезпечує механізмів достовірної перевірки збережених даних або валідації заявлених

облікових записів. Це перетворює такі платформи на аналог цифрової коробки з паперовими сертифікатами – без підтвердженої достовірності та юридичної сили.

Ефективне рішення цієї проблеми пропонує блокчейн через впровадження перевірених смарт-контрактів. У момент додавання досягнень до системи, інформація зберігається у блокчейн-ланцюгу та перевіряється мережею вузлів. Після досягнення порогу підтверджень з боку авторитетних користувачів досягнення отримує індекс довіри, що відображає його достовірність. Подібні проєкти вже впроваджуються у сфері EdTech.

1.4.2 Блокчейн як спосіб перевірки акредитації

У Європі існують сотні різних систем акредитації. Щоб встановити легітимність диплома, необхідно перевірити:

- чи дійсно навчальний заклад видав документ;
- чи має акредитацію відповідний орган;
- чи сам акредитаційний орган має відповідні повноваження.

Усі ці кроки вимагають значних ресурсів. Блокчейн дає змогу зберігати публічні записи про акредитації, підписи органів у відкритому реєстрі, що значно спрощує перевірку походження та валідності освітніх документів.

1.4.3 Блокчейн для відстеження та захисту інтелектуальної власності

Інтелектуальна власність (ІВ) – це основа сучасної творчої економіки. Проте автори, митці, науковці та розробники часто стикаються з проблемами підтвердження свого авторства, реєстрації прав та захисту від несанкціонованого використання своїх робіт. Традиційна система управління ІВ вимагає участі посередників, складної бюрократії та дорогих юридичних процедур. В умовах цифрової епохи, коли інформація швидко копіюється та поширюється, виникає потреба в новій моделі довіри та контролю, яку може

забезпечити блокчейн.

Серед основних проблем у сфері інтелектуальної власності можна виділити відсутність ефективних механізмів для доведення авторства, складність відстеження використання творів у цифровому просторі, проблеми, що виникають при видачі ліцензій третім сторонам, неможливість швидкого виявлення та припинення порушень, а також обмежений доступ до офіційних реєстрів та значні витрати на юридичний захист.

Блокчейн-технології дозволяють ефективно фіксувати дату створення твору, його вміст та власника через криптографічне хешування та смарт-контракти. Це автоматично створює незмінний цифровий запис про виникнення авторських прав відразу після створення об’єкта. Усі подальші операції, такі як зміна власника, видача ліцензій, підтвердження права на використання, також можуть бути записані в блокчейн, створюючи надійний та повний ланцюг володіння.

Прикладами успішного застосування блокчейн-технологій для захисту інтелектуальної власності є американська платформа Binded, яка дозволяє авторам цифрових зображень фіксувати свої права та отримувати сертифікати автентичності, а також здійснювати моніторинг використання контенту через інтеграцію з соціальними мережами та сервісами Google. Німецький проєкт Ascribe пропонує аналогічні послуги художникам, дизайнерам та фотографам, забезпечуючи контроль над доступом до ліцензій. Po.et (США) є блокчейн-протоколом, розробленим для видавництв та журналістів, що дозволяє фіксувати авторські права на тексти та медіаконтент. Ujo Music, також зі США, створює умови для музикантів публікувати свої треки, отримувати роялті напряму від слухачів і надійно фіксувати своє авторство на платформі Ethereum.

Переваги застосування блокчейну включають усунення необхідності в посередниках завдяки автоматичному підтвердженню прав через блокчейн, миттєву реєстрацію творів без звернення до державних органів, глобальну доступність для верифікації у будь-якій країні світу, ефективний контроль

повторного використання через автоматичні ліцензії на основі смарт-контрактів та застосування штучного інтелекту для виявлення порушень у цифровому просторі.

Таким чином, блокчейн виступає не лише як інструмент фіксації авторських прав, але й як потужний механізм автоматичного правозастосування. Це дозволяє створити нові моделі винагородження авторів за використання їхніх творів, подібні до систем цитування в наукових публікаціях, що відкриває широкі можливості для розвитку відкритих освітніх ресурсів (OER) з ефективним правовим контролем. У підсумку, блокчейн пропонує нові шляхи захисту, верифікації, монетизації та контролю інтелектуальної власності у глобальному цифровому середовищі.

1.4.4 Використання для ідентифікації студентів

Використання блокчейн-технології дозволяє створити децентралізовану систему ідентифікації. Після внесення персональних даних студентом до системи, він отримує ідентифікаційну картку у вигляді криптографічного ключа. У поєднанні з біометрією цей ключ дозволяє студенту ідентифікувати себе без необхідності зберігання або передачі особистої інформації.

Переваги:

- студент залишається єдиним власником своїх даних;
- організації не зберігають ПД, а лише підтверджують ідентичність;
- спрощення процедур автентифікації в гуртожитках, бібліотеках, їдальнях;
- зниження навантаження на IT-інфраструктуру та вимог до персоналу.

Це створює безпечну, конфіденційну та масштабовану систему цифрової ідентичності для освітнього середовища.

1.5 Підсумки до першого розділу

Наразі використання блокчейн-технологій в освіті перебуває переважно на експериментальному етапі, проте спостерігається зростаючий інтерес до їхнього впровадження в різноманітні освітні процеси. Такі інновації як цифрова ідентифікація студентів, токенізація сертифікатів, децентралізоване зберігання дипломів та оцінок поступово стають предметом практичного дослідження та апробації в закладах вищої освіти по всьому світу. Прикладами такого впровадження є ініціативи Massachusetts Institute of Technology (MIT) та Університету Нікосії (Кіпр), які вже реалізували пілотні проєкти з видачі дипломів на основі блокчейн-технологій.

У ході аналізу було виявлено значний потенціал використання блокчейн-рішень в освітній галузі. Серед основних переваг – забезпечення автентичності та незмінності освітніх записів, підвищення рівня довіри між учасниками освітнього процесу, можливість зменшення бюрократичного навантаження, створення цифрового портфоліо досягнень студента, а також глобальна доступність і перевірка освітніх даних. Блокчейн також дозволяє запровадити механізми персоналізованого навчання, сприяє розвитку відкритих освітніх ресурсів та створює інфраструктуру для ефективного захисту інтелектуальної власності у сфері освіти.

Однак, не зважаючи на очевидні переваги, впровадження блокчейн-технологій супроводжується низкою викликів. Це і правові невизначеності, і складність масштабування систем, і обмежена технічна обізнаність персоналу освітніх закладів, і необхідність значних інвестицій в інфраструктуру та інтеграційні процеси. Також наявні ризики конфіденційності, що вимагають додаткових технічних і організаційних рішень для їхнього усунення.

2 ДОСЛІДЖЕННЯ ЗАСОБІВ ТОКЕНІЗАЦІЇ ОСВІТНІХ АКТИВІВ НА ОСНОВІ ТЕХНОЛОГІЇ БЛОКЧЕЙНУ ETHEREUM

2.1 Поняття блокчейну та його види

Блокчейн – це розподілений цифровий реєстр (distributed ledger), який дозволяє зберігати інформацію у вигляді ланцюга взаємопов’язаних блоків без потреби в централізованому органі управління. Кожен блок містить набір транзакцій або даних, а також криптографічні посилання на попередній блок, що забезпечує цілісність і незмінність усього ланцюга.

Блокчейн-технологія забезпечує децентралізоване зберігання даних, при якому інформація розподілена між великою кількістю вузлів по всьому світу. Це дозволяє уникнути залежності від єдиного центрального сервера. Водночас, система гарантує прозорість, оскільки всі записи в публічному блокчейні відкриті для перевірки будь-яким користувачем. Ще однією ключовою властивістю є незмінність: після того, як запис потрапив до блокчейну, його не можна змінити або видалити заднім числом. Безпека даних досягається за рахунок використання криптографічних алгоритмів та спеціальних механізмів консенсусу, які запобігають несанкціонованим змінам.

Блокчейн-система має кілька основних структурних елементів. Першим з них є блоки. Кожен блок містить список транзакцій, мітку часу, хеш поточного блоку, хеш попереднього блоку та спеціальний параметр nonce, що особливо актуальний у системах, які використовують алгоритм Proof of Work.

Наступним компонентом є ланцюг блоків, де кожен блок пов’язаний із попереднім у суворій хронологічній послідовності. Будь-яка зміна в одному з попередніх блоків призведе до зміни його хешу, що, своєю чергою, зруйнує цілісність усіх наступних блоків і зробить їх недійсними. Саме ця властивість

забезпечує незмінність і надійність усього ланцюга даних.

Хеш-функції відіграють критично важливу роль у верифікації та цілісності інформації. Для кожного блоку обчислюється унікальний хеш, що залежить від його вмісту. Типовими прикладами таких функцій є SHA-256 і Keccak-256, які забезпечують однозначність і захист даних від змін.

Ще одним важливим елементом є мережа вузлів або нодів. Усі вузли зберігають повну копію блокчейну та беруть участь у перевірці транзакцій. Залежно від функцій, які вони виконують, вузли можуть бути повними, легкими, а також діяти як майнери або валідатори.

Останнім компонентом, який визначає роботу системи, є протокол консенсусу. Існує кілька типів таких протоколів. Алгоритм Proof of Work використовується, зокрема, у Bitcoin і базується на витраті обчислювальних ресурсів. У мережі Ethereum 2.0 впроваджено Proof of Stake, який передбачає вибір валідаторів на основі кількості заблокованих токенів. Інші варіанти включають Delegated Proof of Stake, який застосовується в мережах EOS та TRON, Proof of Authority, характерний для приватних блокчейнів, а також нові гібридні алгоритми, зокрема Proof of History (використовується в Solana) та Proof of Reputation. Всі ці підходи розглядаються у таблиці 2.1.

Таблиця 2.1 – Протокол консенсусу

Тип блокчейну	Характеристика	Приклади
Публічний (Public)	Відкритий для всіх, будь-хто може стати учасником мережі.	Bitcoin, Ethereum
Приватний (Private)	Обмежений доступ, учасники відомі, керується однією організацією.	Hyperledger Fabric
Консорціум (Consortium)	Кілька організацій спільно керують мережею.	R3 Corda, Quorum
Гібридний (Hybrid)	Поєднання публічних і приватних елементів.	Dragonchain, XinFin

Публічний (відкритий) блокчейн (Public Blockchain) є найбільш поширеним типом мережі, в якій будь-який користувач має можливість приєднатися до системи, зчитувати й записувати дані, а також брати участь у процесі валідації транзакцій, наприклад через майнінг або стейкінг. Цей тип блокчейну повністю децентралізований, користувачі не зобов'язані розкривати свою особистість, а всі транзакції залишаються публічними та прозорими. Основними перевагами є повна відкритість, стійкість до цензури та високий рівень довіри, що досягається завдяки відкритому механізму валідації. Водночас система має і недоліки – зокрема, повільну швидкість обробки транзакцій у традиційних PoW-системах, значне енергоспоживання та неможливість обмежити доступ до інформації. Прикладами таких мереж є Bitcoin, що є першою реалізацією публічного блокчейну для криптовалюти BTC, а також платформа Ethereum, призначена для створення смарт-контрактів і децентралізованих додатків. До новітніх реалізацій цього типу належать Solana, Polkadot і Cardano – платформи покоління 3.0–4.0.

Приватний блокчейн (Private Blockchain) являє собою закриту мережу з обмеженим доступом, якою користуються лише певні організації або зареєстровані користувачі. Всі учасники ідентифіковані, а право на додавання нових блоків належить лише авторизованим вузлам, що перебувають під контролем однієї організації. Цей підхід забезпечує високу швидкість транзакцій, знижене енергоспоживання, можливість контролю доступу до даних та збереження конфіденційності. Однак така система втрачає децентралізований характер, має обмежену прозорість і менший рівень довіри з боку зовнішніх користувачів. Як приклад можна навести Hyperledger Fabric – корпоративну платформу від Linux Foundation, та Corda – систему, орієнтовану на банківські й фінансові установи.

Консорціумний блокчейн (Federated або Consortium Blockchain) відрізняється тим, що управління ним здійснюється спільно кількома організаціями. Це дозволяє поєднати переваги як публічного, так і приватного підходів, коли дозволи на читання та запис даних визначаються

групою вузлів. Така структура забезпечує розподілене управління, що підвищує рівень довіри, забезпечує баланс між прозорістю та приватністю, а також дозволяє досягати високої масштабованості. Основними недоліками є складність управління мережею, яка вимагає координації між усіма сторонами, та обмежений рівень публічного доступу. До прикладів належать Quorum – корпоративна версія Ethereum, розроблена компанією J.P. Morgan, Energy Web Chain – проєкт у сфері енергетики, а також IBM Food Trust – платформа для відстеження поставок у харчовій промисловості.

Гібридний блокчейн (Hybrid Blockchain) поєднує у собі елементи як публічного, так і приватного блокчейну. Частина інформації в таких системах зберігається відкрито, тоді як інші дані залишаються закритими, що забезпечує гнучкість у регулюванні доступу та вибіркову відкритість. Серед переваг – можливість налаштовувати рівні доступу для різних користувачів, висока продуктивність роботи системи, а також підтримка функцій конфіденційності. Водночас архітектура таких систем є більш складною, а деякі частини можуть залишатися під контролем центрального регулятора. Прикладами гібридних систем виступають Dragonchain – платформа, розроблена компанією Disney, та XinFin (XDC) – рішення для бізнесу та державного використання. Типову структуру кожного типа блокчейну показані на рисунку 2.1.

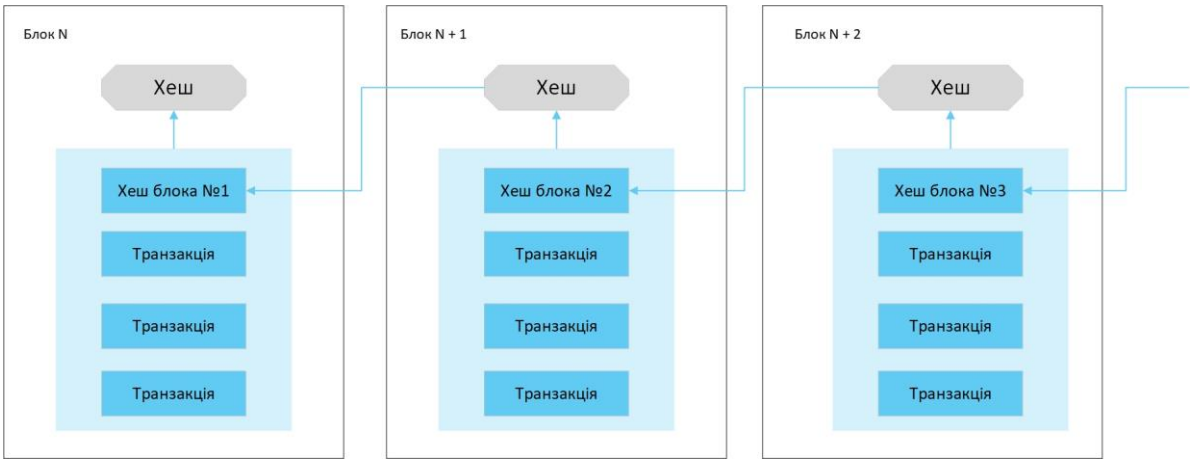

Рисунок 2.1 – Типова структура блокчейну

Блокчейн як інноваційна технологія має низку ключових переваг, які забезпечують її ефективне використання в різних сферах: від фінансів і логістики до охорони здоров'я, освіти та захисту інтелектуальної власності. Однією з таких переваг є прозорість, адже всі учасники мережі мають доступ до однакової, синхронізованої версії блокчейну. Це дозволяє незалежно перевіряти будь-які транзакції, зміни або події в системі, зменшуючи ризик корупції, шахрайства та маніпуляцій. Прикладом може слугувати платформа IBM Food Trust, яка дає змогу прозоро відстежувати ланцюг постачання продуктів – від виробника до споживача.

Ще однією важливою характеристикою є безпека. Завдяки використанню криптографічних алгоритмів, таких як SHA-256 чи ECDSA, а також децентралізованій структурі, блокчейн має високу стійкість до зовнішніх атак. У разі порушення одного вузла вся інша мережа залишається захищеною. Наприклад, у системі Bitcoin зміна інформації в одному блоці вимагає перерахунку всіх наступних, що практично неможливо без контролю над більшістю обчислювальних потужностей мережі.

Незмінність даних також є ключовою властивістю: після запису та підтвердження даних у блокчейні, їх не можна змінити або видалити без згоди більшості учасників мережі. Це гарантує надійне збереження історії подій, що особливо актуально при верифікації сертифікатів або медичних записів.

Децентралізація є ще однією визначальною характеристикою блокчейну. Відсутність централізованого управління знижує ризики зловживання владою чи технічних збоїв. Кожен вузол мережі зберігає копію реєстру, а будь-які зміни можливі лише за умови досягнення згоди між усіма учасниками.

Автоматизація процесів за допомогою смарт-контрактів значно підвищує ефективність систем. Смарт-контракти – це самовиконувані програми, що автоматично реалізують умови угоди після виконання визначених дій. Вони дозволяють виключити посередників, пришвидшити

транзакції та зменшити витрати на аудит. Наприклад, на платформі Ethereum можна створити смарт-контракт для автоматичної видачі токенізованого диплому після проходження курсу.

Використання блокчейн-технологій сприяє зниженню витрат і підвищенню ефективності завдяки усуненню зайвих ланок у ланцюгах обміну інформацією та автоматизації звітності. Це дозволяє зменшити операційні витрати і прискорити обробку. У сфері логістики, наприклад, блокчейн допомагає уникнути дублювання документів, пришвидшує митні процедури та мінімізує помилки.

Важливою є також властивість відстежуваності та трасованості даних. Блокчейн дозволяє фіксувати кожну транзакцію або зміну з точністю до часу й дії. Це створює високий рівень довіри до даних у таких сферах, як освіта, фінанси, медицина та логістика. Система OpenCerts у Сінгапурі, наприклад, забезпечує перевірку достовірності дипломів шляхом простежування ланцюга їхньої видачі.

Гнучкість у налаштуванні конфіденційності та контролю доступу є ще однією перевагою, особливо для приватних і гібридних блокчейнів. Такі системи дозволяють налаштовувати рівень доступу до різних типів інформації, забезпечуючи як конфіденційність, так і прозорість. Наприклад, у Hyperledger Fabric можна забезпечити доступ до певних даних лише для визначених організацій, навіть якщо вони є частиною однієї мережі.

2.2 Хешування у блокчейні

2.2.1 Криптографія

Криптографія є фундаментальною технологією, що забезпечує безпеку, надійність та довіру до блокчейн-систем. Вона дозволяє захистити дані від несанкціонованого доступу, гарантувати цілісність записів і здійснювати верифікацію транзакцій без посередників. У контексті блокчейн-технологій

криптографія виконує низку критичних функцій, серед яких захист транзакцій, аутентифікація користувачів, забезпечення цілісності даних і підтримка смарт-контрактів.

Захист транзакцій досягається завдяки шифруванню і підписуванню даних криптографічними ключами. Це унеможливлює зміну або підробку вмісту транзакції без знання приватного ключа її власника. Користувачі мережі ідентифікуються за допомогою унікальних криптографічних пар ключів – приватного (секретного) та публічного (відкритого). Приватний ключ використовується для підпису транзакцій, тоді як публічний – для їх перевірки, що забезпечує автентичність користувача без необхідності централізованої перевірки.

Цілісність даних у блоках блокчейну забезпечується хеш-функціями, які створюють унікальний цифровий відбиток (хеш) на основі вмісту блоку. Навіть найменша зміна в блоці призводить до повної зміни хешу, що робить фальсифікацію фактично неможливою. У випадку смарт-контрактів криптографічні методи гарантують їх безпечне виконання. Такі контракти автоматично реалізовують логіку угод між сторонами, виключаючи необхідність у третіх особах.

У блокчейн-системах застосовується кілька базових криптографічних технологій. Асиметричне шифрування базується на використанні пари ключів: приватного для підпису й публічного для перевірки. Алгоритм ECDSA (Elliptic Curve Digital Signature Algorithm) є прикладом такого підходу і використовується в Bitcoin та Ethereum для створення цифрових підписів. Цифровий підпис дозволяє підтвердити, що транзакція справді створена власником адреси, і навіть незначні зміни в підписаному повідомленні призводять до провалу перевірки підпису. Хеш-функції, зокрема SHA-256 (використовується в Bitcoin) і Keccak-256 (застосовується в Ethereum), створюють короткий унікальний рядок з вхідних даних, який унеможливлює відновлення оригінального змісту, забезпечуючи при цьому однозначну верифікацію даних.

Хешування – це один із базових механізмів, що забезпечує безпеку, незмінність і цілісність даних у блокчейн-мережах. У найзагальнішому вигляді хеш-функція виконує перетворення вхідного повідомлення довільного розміру на рядок фіксованої довжини (хеш або дайджест), який є унікальним цифровим відбитком цієї інформації.

У блокчейн-технології хеш-функції застосовуються для:

- побудови зв’язків між блоками (через хеш попереднього блоку);
- перевірки цілісності транзакцій (меркле-дерева);
- пошуку блоків у системах Proof of Work (через обчислення nonce);
- створення цифрових підписів та ідентифікації користувачів.

При створенні нового блоку в блокчейн-мережі його вміст, який включає транзакції, мітку часу, nonce і хеш попереднього блоку, передається через хеш-функцію. Отриманий результат – це хеш нового блоку, який далі фіксується в наступному блоці, створюючи ланцюгову зв’язаність і забезпечуючи незмінність усієї історії транзакцій. Хеш-функції мають декілька важливих математичних і криптографічних властивостей, які роблять їх ідеальними для використання у блокчейні.

По-перше, вони детерміновані, тобто для одних і тих самих вхідних даних результат хешування завжди буде однаковим, що дозволяє точно перевіряти транзакції за хешами.

По-друге, вони мають високу швидкість обчислення, що дозволяє ефективно працювати у режимі реального часу, наприклад, у системах майнінгу. Також важливою є односторонність – неможливість зворотного обчислення початкових даних із хешу, що забезпечує конфіденційність.

Крім того, хеш-функції вирізняються колізійною стійкістю – ймовірність того, що дві різні послідовності вхідних даних утворять однаковий хеш, є надзвичайно низькою. Нарешті, вони демонструють ефект лавини: навіть мінімальна зміна у вхідних даних призводить до кардинальної зміни хешу, що дозволяє легко виявляти спроби підробки або модифікації блоків чи транзакцій.

2.2.2 DApp (Децентралізовані додатки)

DApp (Decentralized Application) – це децентралізований програмний додаток, що працює на базі блокчейн-мережі, а не на централізованому сервері або традиційній клієнт-серверній архітектурі. Такі додатки створюються для забезпечення прозорої, захищеної, безпечної та стійкої до цензури взаємодії між користувачами. На відміну від звичних веб або мобільних застосунків, DApp характеризуються децентралізацією, реалізацією логіки за допомогою смарт-контрактів, відкритим вихідним кодом і використанням токенів.

Децентралізація означає, що як код, так і дані зберігаються у розподіленій мережі вузлів, що унеможливлює централізоване управління або втручання з боку окремої особи чи організації. Основна логіка роботи таких застосунків реалізується за допомогою смарт-контрактів, які автоматично виконуються при виконанні заданих умов і не потребують посередників. Відкритість коду забезпечує прозорість і можливість аудиту з боку спільноти, що сприяє зростанню довіри до системи. У багатьох DApp реалізовано власну токеноміку, де токени використовуються для оплати послуг, винагород, голосування або участі в управлінні проєктом через децентралізовані автономні організації (DAO).

Переваги децентралізованих застосунків полягають у тому, що вони усувають потребу в посередниках – користувачі взаємодіють безпосередньо, що дозволяє знизити комісії та пришвидшити обробку транзакцій. Безпека і захист даних досягаються завдяки використанню криптографії: усі персональні дані і транзакції захищені та не можуть бути змінені або видалені без загальної згоди мережі. Смарт-контракти дозволяють автоматизувати виконання угод, знижуючи ризик помилок, пов’язаних із людським фактором. Крім того, DApp характеризуються високою стійкістю до цензури, оскільки не мають єдиного центру обробки – їх важко заблокувати або атакувати. Уся діяльність у таких застосунках є прозорою –

кожна транзакція або зміна логіки контракту зберігається в блокчейні і може бути перевірена будь-яким учасником.

Разом із тим, DApp мають і певні недоліки. Їх розробка є технічно складною і вимагає знань з мов програмування смарт-контрактів (наприклад, Solidity), роботи з бібліотеками Web3, а також розуміння криптографічних принципів і особливостей мереж Ethereum або альтернативних Layer 2-рішень. Масштабування також становить проблему: публічні блокчейни мають обмежену пропускну здатність, що може спричинити затримки під час обробки великої кількості транзакцій, особливо у періоди пікових навантажень. Крім того, через децентралізовану архітектуру деякі DApp мають менш інтуїтивний інтерфейс і можуть працювати повільніше порівняно з традиційними централізованими додатками. Типи існуючих DApp показані в таблиці 2.2.

Таблиця 2.2 – Типи DApp

Тип DApp	Характеристика	Приклади
DeFi (фінансові)	Кредитування, обмін, стейкінг	Uniswap, Aave, MakerDAO
Ігрові (GameFi)	Ігри з NFT, Play-to-Earn	Axie Infinity, Decentraland
Соціальні	Децентралізовані соцмережі	Lens Protocol, Minds
Маркетплейси	Торгівля NFT, цифровими активами	OpenSea, Rarible
Освітні	Збереження сертифікатів, токенізація знань	OpenCerts, EduChain

DApp у сфері децентралізованих фінансів (DeFi) є одним із найрозвиненіших напрямів блокчейн-екосистеми. Такі додатки надають фінансові послуги без залучення посередників, таких як банки чи брокери, що дозволяє знизити вартість операцій і підвищити їхню доступність. Серед

типових функцій DeFi DApp можна виокремити кредитування та позики, стейкінг і фармінг, децентралізовані біржі, страхування на блокчейні, а також алгоритмічні стейблкоїни. До прикладів належать Uniswap – децентралізована біржа на Ethereum, Aave – платформа для кредитування активів з динамічною процентною ставкою, і Compound – протокол управління позиками через токени типу cDAI та cUSDC.

Ігрові DApp, що належать до категорій GameFi та Play-to-Earn, поєднують у собі елементи блокчейн-технологій і гейміфікації. Вони дозволяють користувачам володіти внутрішньоігровими активами у вигляді NFT, здійснювати обмін цими активами та отримувати криптовалюту за участь у грі. Характерними рисами таких додатків є використання унікальних токенізованих предметів, економіка винагород за активність, а також наявність відкритих вторинних ринків для торгівлі активами. Серед прикладів можна згадати Axie Infinity, де користувачі можуть битися, вирощувати і продавати NFT-персонажів, Decentraland – віртуальний світ з можливістю володіння нерухомістю на блокчейні, та Gods Unchained – карткову гру, де кожна карта є NFT.

Соціальні DApp, відомі також як SocialFi або Web3 Social, розробляються як децентралізовані соціальні мережі, в яких користувачі зберігають повний контроль над власними даними. Учасники таких платформ можуть отримувати винагороди за створення контенту, взаємодію із спільнотою або участь у голосуваннях. Ці додатки забезпечують захист особистої інформації, монетизацію без посередників і децентралізоване управління спільнотами через DAO. До прикладів належать Lens Protocol, який використовується для створення Web3-соцмереж, Minds – соціальна мережа з криптовалютною винагородою, а також Farcaster – протокол для інтеграції між Web3-соціальними застосунками.

Ринкові DApp, або Marketplace DApps, надають користувачам платформи для купівлі, продажу або обміну цифровими активами без посередників. Вони функціонують на основі peer-to-peer взаємодії, з

підтримкою торгівлі NFT, токенізованим мистецтвом, музикою чи доменами. До найвідоміших ринкових DApp відносяться OpenSea – найбільший NFT-маркетплейс, Rarible – платформа для художників і колекціонерів з власною токеномікою RARI, а також LooksRare – конкурент OpenSea, який акцентує увагу на спільнотній взаємодії.

Освітні DApp, або EdTech-додатки на блокчейні, застосовуються для зберігання, підтвердження та обміну освітніми досягненнями. Вони дозволяють токенізувати сертифікати, дипломи, результати курсів і навички, забезпечуючи незмінність, достовірність та автономію студентів у володінні власним навчальним треком. Прикладами таких застосунків є OpenCerts у Сінгапурі – державна ініціатива цифрових дипломів, EduChain – платформа для токенізації сертифікатів, і BCDiploma – сервіс для створення верифікованих документів на Ethereum.

DApp для ідентифікації, або додатки концепції Self-Sovereign Identity, дозволяють користувачам самостійно управляти своєю цифровою ідентичністю без централізованих баз даних. Завдяки таким рішенням, користувачі самі визначають, хто і коли може мати доступ до їхніх персональних даних. Серед таких прикладів можна назвати uPort – ідентифікаційний протокол на Ethereum, Civic – додаток для верифікації особистості та безпарольного входу, а також Serto – рішення, побудоване на технології децентралізованих ідентифікаторів (DID).

2.2.3 SHA-256

SHA-256 (Secure Hash Algorithm 256-bit) – це криптографічна хеш-функція, яка перетворює вхідні дані довільної довжини у фіксоване 256-бітне (32-байтне) вихідне значення, яке також називають дайджестом або хеш-кодом. Алгоритм розроблений Національним інститутом стандартів і технологій США (NIST) і входить до складу сімейства алгоритмів SHA-2, що також включає варіанти SHA-224, SHA-384 та SHA-512.

SHA-256 має низку ключових характеристик. По-перше, незалежно від розміру вхідних даних, функція завжди генерує 256-бітний хеш. Це забезпечує зручність порівняння даних за хешами без потреби звертатись до оригінальних значень. Другою важливою властивістю є детермінованість, яка означає, що однакові вхідні дані завжди дають однаковий результат. Односторонність алгоритму полягає в тому, що неможливо або практично неможливо відновити оригінальні дані, знаючи лише хеш, що гарантує конфіденційність.

Крім того, SHA-256 стійкий до колізій, тобто ймовірність того, що два різні вхідні значення створять однаковий хеш, є надзвичайно низькою. Ще однією ключовою властивістю є чутливість до змін у вхідних даних: навіть мінімальна зміна, наприклад, одного біта, призведе до радикально іншого хешу. Цей ефект, відомий як лавинний, особливо важливий для забезпечення надійності криптографічних систем.

Процес хешування в SHA-256 включає кілька послідовних етапів. Спочатку здійснюється попередня обробка, або padding, коли вхідні дані доповнюються до розміру, який кратний 512 бітам. Для цього додається одиничний біт, потім нулі, щоб заповнити до 448 бітів, а останні 64 біти використовуються для зберігання довжини початкового повідомлення. Це гарантує, що загальний розмір буде кратним 512 бітам.

Далі виконується ініціалізація буферів. Алгоритм використовує вісім 32-бітних регістрів, які загалом охоплюють 256 бітів, і ці регістри ініціалізуються константами, що походять від дробових частин квадратних коренів перших восьми простих чисел. Потім йде обробка блоків повідомлення: вхідні дані розбиваються на блоки по 512 бітів, і кожен блок проходить 64 ітерації основного циклу компресії. У цьому циклі виконуються логічні та арифметичні операції, зокрема додавання, XOR, циклічні та побітові зсуви.

Основний цикл включає функцію компресії, яка змішує вміст регістрів з метою усунення статистичних залежностей між вхідними даними та

кінцевим хешем. Після кожного блоку регістри оновлюються, що забезпечує вплив усіх блоків на кінцевий результат. Наприкінці усі значення регістрів об'єднуються для формування кінцевого 256-бітного хешу.

Серед переваг SHA-256 варто виділити високу безпеку – він не має відомих вразливостей на момент написання, що робить його надійним для зберігання паролів, цифрових підписів і чутливих даних. Він активно використовується в блокчейн-технологіях, зокрема в системі Bitcoin. Тут він є основним алгоритмом хешування, що використовується для створення унікальних ідентифікаторів блоків і транзакцій, а також для захисту від змін у блокчейні.

SHA-256 також є відносно швидким і забезпечує ефективний баланс між швидкістю та рівнем безпеки, що робить його популярним у багатьох прикладних сферах. Він широко використовується не тільки в блокчейні, але й у криптографічних протоколах, таких як SSL/TLS для захищених інтернет-з'єднань, або PGP для шифрування електронної пошти.

У контексті майнінгу блокчейну, як у Bitcoin, SHA-256 відіграє центральну роль. Майнер повинен знайти таке значення nonce, щоб хеш заголовка блоку, який включає транзакції та інформацію про попередній блок, відповідав певному критерію – наприклад, починався з визначеної кількості нульових бітів. Якщо такий хеш знайдено, блок визнається дійсним і додається до блокчейну.

Завдяки своїм властивостям – односторонності, стійкості до колізій, лавинному ефекту – SHA-256 залишається ключовим криптографічним інструментом у сучасних цифрових системах і критично важливим елементом для безпеки в екосистемі блокчейну. Обробку блоку можна побачити на рисунку 2.2.

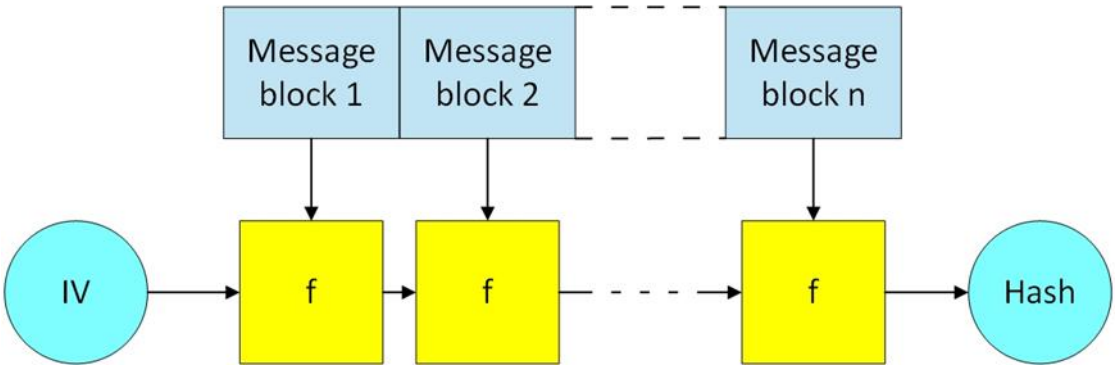

Рисунок 2.2 – Обробка блоку у SHA-256

2.2.4 Ethash

Ethash – це алгоритм консенсусу типу proof-of-work (PoW), який застосовувався в блокчейн-мережі Ethereum до переходу на Ethereum 2.0. Ethash був спеціально розроблений для забезпечення високого рівня безпеки, протидії централізації та збереження рівного доступу до майнінгу через використання графічних процесорів (GPU).

Подібно до інших PoW-алгоритмів, Ethash вимагає від майнерів виконати обчислювальну роботу – знайти значення (nonce), яке дозволить згенерувати хеш нового блоку, що задовольняє певні умови складності. Проте Ethash має низку унікальних характеристик, які виділяють його серед інших алгоритмів, наприклад, SHA-256 у Bitcoin.

Однією з головних особливостей Ethash є його орієнтованість на GPU-майнінг. Алгоритм спроектований так, щоб бути неефективним для ASIC-пристроїв, які домінують у мережі Bitcoin. Це сприяє демократизації майнінгу, оскільки звичайні користувачі можуть добувати монети за допомогою доступних відеокарт. Крім того, Ethash має високі вимоги до пам’яті. Він є memory-hard, тобто потребує значного обсягу оперативної пам’яті для генерації та обробки даних. Це ускладнює розробку спеціалізованих чіпів (ASIC), які були б ефективнішими за GPU.

Ще однією ключовою характеристикою Ethash є використання DAG-файлу (Directed Acyclic Graph). Для роботи Ethash необхідно згенерувати

великий набір даних – DAG-файл, який оновлюється кожні 30 000 блоків (приблизно кожні 5–6 днів) та з часом постійно зростає. Цей файл є ключовим елементом хешування і має бути завантажений у пам’ять відеокарти. Розмір DAG-файлу є важливим чинником, що впливає на сумісність і продуктивність майнерів.

Ethash також відзначається складністю обчислення та легкістю перевірки. Алгоритм розрахований так, що знайти правильний хеш складно, але перевірити його коректність – легко. Це важливо для підтримання швидкого обміну блоками між вузлами мережі.

Ethash був основним алгоритмом майнінгу в мережі Ethereum з моменту її запуску у 2015 році до 2022 року, коли відбувся перехід до Ethereum 2.0 і було реалізовано алгоритм консенсусу Proof of Stake (PoS). Замість обчислювального майнінгу мережа перейшла до стейкінгу, де валідатори підтверджують транзакції, розміщуючи ETH як заставу.

Ethash став знаковим прикладом PoW-алгоритму, орієнтованого на децентралізацію майнінгу та доступність для широкого кола учасників. Його використання в Ethereum відіграло важливу роль у становленні екосистеми смарт-контрактів та децентралізованих додатків. Перехід до Proof of Stake відбувся з міркувань масштабованості, екологічності та енергозбереження, однак Ethash залишається важливою частиною історії блокчейн-технологій. Хід роботи алгоритму хешування Ethash можливо узагальнити як показано на рисунку 2.3.

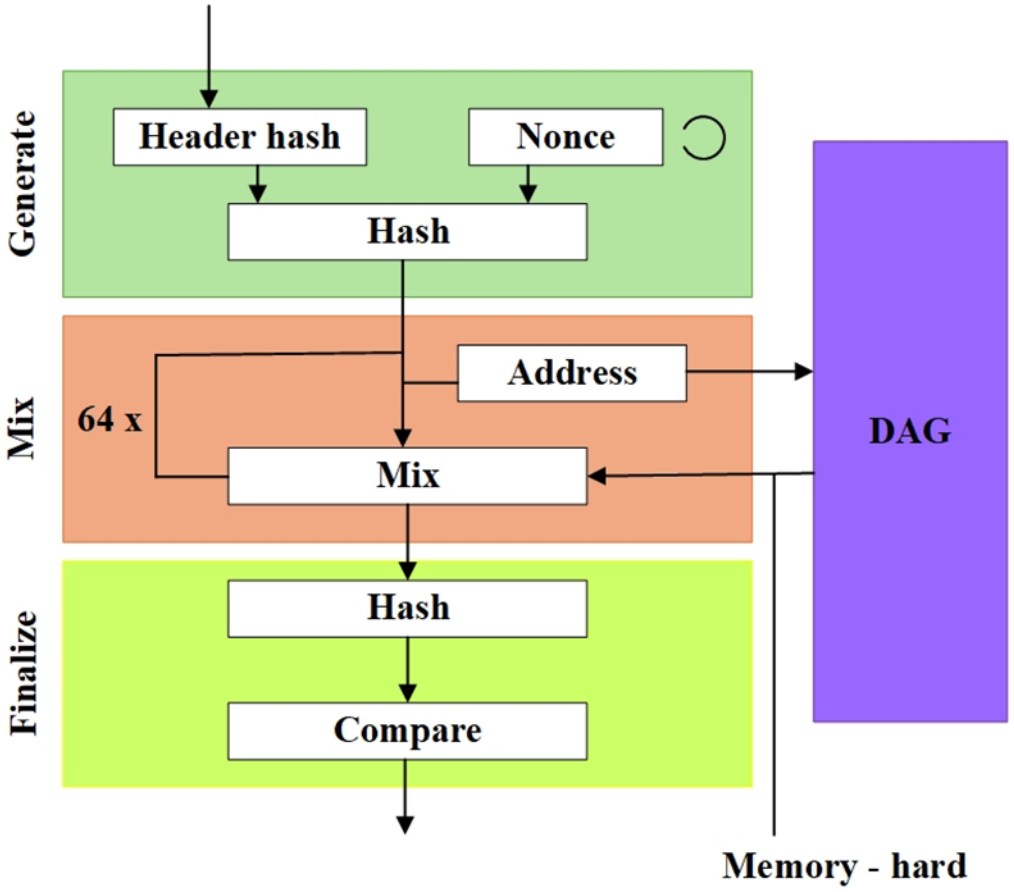

Рисунок 2.3 – Процес роботи алгоритму Ethash

2.3 Ethereum

Ethereum – це відкрита децентралізована блокчейн-платформа, що надає функціональність створення та розгортання смарт-контрактів і децентралізованих додатків (DApps). Запущена у 2015 році, вона є другою за капіталізацією криптовалютою у світі після Bitcoin, але, на відміну від нього, Ethereum не обмежується лише фінансовими транзакціями. Його головна інновація – Ethereum Virtual Machine (EVM), яка дозволяє будь-кому створювати програмований блокчейн-код, що виконується децентралізовано.

EVM – це середовище виконання смарт-контрактів. Воно є однаковим для всіх нод мережі, що гарантує узгоджене виконання коду. Програми для EVM пишуться мовою Solidity або Vyper, компілюються у байт-код та виконуються на кожному вузлі мережі.

Смарт-контракт – це автономна програма, яка автоматично виконується

після виконання заданих умов. Вони є незмінними після розгортання, відкритими для перегляду у блокчейні та дозволяють створювати токени, DAO, децентралізовані біржі, NFT та інші децентралізовані сервіси.

Ethereum дозволяє створювати власні цифрові активи (токени), що працюють на його базі. Найпоширенішими стандартами є ERC-20 для взаємозамінних токенів, як-от DAI, USDC, UNI; ERC-721 для унікальних NFT, наприклад, CryptoKitties або Bored Apes; ERC-1155 – комбінований стандарт для ігрових предметів та інших активів.

Ethereum використовує механізм gas для обмеження обчислень та стимулювання майнерів або валідаторів. Чим складніше обчислення, тим більше gas необхідно, і тим вища плата за транзакцію.

Спочатку Ethereum використовував механізм Proof of Work (PoW) на базі алгоритму Ethash, що забезпечував безпеку, але споживав багато енергії. Для подолання обмежень масштабованості та екологічних викликів було реалізовано Ethereum 2.0 – поетапний перехід до Proof of Stake (PoS).

Основними змінами Ethereum 2.0 стали запуск Beacon Chain як основи нової PoS-мережі, яка координує валідаторів і слоти. Замість майнінгу користувачі блокують ETH (мінімум 32 ETH), щоб стати валідаторами. У вересні 2022 року відбулося злиття (The Merge) PoW-мережі з Beacon Chain, після чого Ethereum остаточно відмовився від PoW.

Метою Ethereum є створення децентралізованої інфраструктури для розробки та запуску додатків і сервісів, які не залежать від централізованих серверів і організацій. Вона надає можливість розробникам створювати програми, які працюють у децентралізованій мережі, що гарантує прозорість, безпеку і стійкість до цензури.

Серед основних переваг Ethereum як платформи варто відзначити її гнучкість, яка дозволяє створювати смарт-контракти з довільною логікою; розвинену екосистему з найбільшою кількістю DApp, токенів, NFT та DeFi-проєктів; інтероперабельність із підтримкою мостів, кросчейн-переказів і стандартів токенів; а також активну спільноту розробників, яка постійно

оновлює платформу за підтримки Ethereum Foundation.

Серед недоліків і викликів Ethereum варто згадати високу комісію за транзакції у пікові періоди, коли плата може сягати десятків доларів; обмежену швидкість обробки – близько 15–30 транзакцій на секунду без застосування рішень другого рівня; і складність платформи для новачків, оскільки створення безпечних смарт-контрактів вимагає глибоких технічних знань.

Щоб подолати технічні обмеження, активно розвиваються Layer 2-рішення, серед яких Arbitrum і Optimism як rollup-платформи для масштабування, ZkSync та StarkNet як zk-rollups для забезпечення приватності й продуктивності, а також Polygon, що функціонує як sidechain, сумісний з Ethereum. Упровадження EIP-4844 (Proto-Danksharding) сприяє подальшому зниженню вартості транзакцій на L2-рішеннях.

Layer 2 (другий рівень) – це технологія або протокол, розміщений поверх основного блокчейну (Layer 1), який дозволяє здійснювати транзакції поза головним ланцюгом із подальшою синхронізацією з ним. Інакше кажучи, L2 переміщує обчислення та зберігання даних поза межі основного ланцюга, а в Layer 1 передає лише підсумкові докази. Транзакції користувачів обробляються на другому рівні, агрегуються (об’єднуються), і тільки зведений результат або криптографічний доказ (наприклад, zk-доказ) надсилається до основного ланцюга Ethereum. Це дозволяє значно скоротити використання gas і зменшити час підтвердження транзакцій.

Основними типами Layer 2-рішень є rollups, sidechains, state channels і validium. Rollups, як найпопулярніший тип, здійснюють транзакції поза ланцюгом, але публікують дані або докази на Layer 1. Optimistic Rollups працюють за принципом припущення про правильність даних із можливістю перевірки у разі суперечок, тоді як Zero-Knowledge Rollups (ZK-Rollups) використовують криптографічні докази, що гарантують правильність обчислень без потреби в довірі.

Sidechains є незалежними блокчейнами з власними механізмами

консенсусу, які взаємодіють з Ethereum через мости, тоді як state channels забезпечують приватні взаємодії між учасниками з публікацією лише початкового та фінального стану. Validium подібний до zk-Rollups, але дані зберігаються поза мережею, що підходить для застосунків з меншою потребою в ончейн-прозорості, таких як ігри чи конфіденційні транзакції.

Перевагами Layer 2 є значне зростання пропускної здатності (до тисяч транзакцій за секунду), зменшення комісій, миттєве підтвердження результатів і збереження безпеки базової мережі Ethereum, особливо у випадку rollups. Водночас викликами залишаються затримки при виведенні активів, технічна складність впровадження в dApp, а також фрагментація інфраструктури через наявність різних токенів, гаманців і несумісних рішень на різних платформах, які ми можемо побачити на таблиці 2.3.

Таблиця 2.3 – Основні платформи Layer 2

Платформа	Тип	TPS (теоретично)	Комісія	Особливості
Optimism	Optimistic Rollup	~2,000+	Низька	Простий перехід з L1
Arbitrum	Optimistic Rollup	~4,500+	Низька	Найбільший TVL серед L2
zkSync Era	ZK-Rollup	~2,000+	Дуже низька	zk-докази для L1 безпеки
StarkNet	ZK-Rollup (STARK)	~3,000+	Дуже низька	Потужна cryptographic інфраструктура
Polygon POS	Sidechain	~7,000+	Дешева	Сумісність із Ethereum, не pure L2

Ethereum Foundation працює над покращенням підтримки L2, зокрема через впровадження EIP-4844 (Proto-Danksharding), який дозволить ефективніше зберігати дані з роллапів на L1, ще більше зменшуючи комісії. Layer 2-рішення – це ключовий етап у масштабуванні Ethereum. Вони дозволяють зберегти безпеку і децентралізацію основної мережі, паралельно підвищуючи її швидкодію і доступність. У контексті освітніх децентралізованих застосунків L2 відкривають можливості для швидкої, дешевої валідації дипломів, сертифікатів, NFT-документів, що робить блокчейн-технології дійсно масовими.

Також у майбутньому очікується уніфікація облікових записів і гаманців, що дозволить користувачам легше взаємодіяти з L2 без необхідності переходити між bridge, токенами gas тощо.

Робимо висновок, що Ethereum – це не лише криптовалюта, а повноцінна платформа для побудови децентралізованої економіки, яка дозволяє створювати прозорі, автономні і безпечні цифрові сервіси. Його розвиток – від PoW до PoS, від монолітної мережі до багаторівневої архітектури з L2-рішеннями – є прикладом еволюції блокчейн-технологій у відповідь на вимоги реального світу.

2.4 Ethereum Classic

Ethereum Classic (ETC) – це блокчейн-платформа, яка зберегла початкову ідеологію Ethereum до хардфорку 2016 року. Вона підтримує смарт-контракти, децентралізовані додатки (DApps) та використовує механізм консенсусу Proof of Work (PoW). Незважаючи на тісну спорідненість з Ethereum, Ethereum Classic має власну філософію розвитку, що базується на принципі "код – це закон" (code is law).

Ethereum Classic виник у результаті суперечки в спільноті Ethereum щодо того, як реагувати на масштабний злом проєкту The DAO у червні 2016 року. В результаті вразливості у смарт-контракті хакер вивів понад 3,6 млн

ETH (~60 млн доларів на той час).

Щоб повернути кошти, команда Ethereum ініціювала хардфорк, який "відмотав" історію блокчейну до моменту злому, і кошти було повернено.

Проте частина спільноти категорично не погодилася з цією дією, вважаючи її порушенням принципу незмінності блокчейну. Вони продовжили розвивати оригінальний ланцюг Ethereum, який і отримав назву Ethereum Classic.

Основний принцип Ethereum Classic – недоторканність блокчейну, навіть якщо це означає втрату коштів. Ідеологія "код – це закон" стверджує, що змінювати ланцюг блоків після його оприлюднення – неприпустимо, навіть у випадку помилки. Це робить Ethereum Classic найпослідовнішим реалізатором децентралізованої ідеї блокчейн-мереж. Представлена характеристика Ethereum Classic у таблиці 2.4.

Таблиця 2.4 – Характеристики Ethereum Classic

Характеристика	Ethereum Classic
Алгоритм консенсусу	Proof of Work (Etchash)
Смарт-контракти	Підтримка Solidity
Мова програмування	Solidity, Vyper
Віртуальна машина	Ethereum Virtual Machine (EVM)
Базовий токен	ETC (Ethereum Classic Coin)
Максимальна емісія	~210 млн ETC (обмежено, як у Bitcoin)
Швидкість блоку	~13 секунд
Комісії за транзакції	Нижчі, ніж в Ethereum

Ethereum Classic довгий час використовував Ethash, однак після переходу Ethereum на PoS у 2022 році було впроваджено Etchash – модифіковану версію Ethash, сумісну з актуальним станом DAG-файлів.

Таким чином, ETC став основним притулком для PoW-майнерів, які вийшли з Ethereum після The Merge. Це частково збільшило хешрейт

Ethereum Classic.

Особливості:

- підтримує GPU-майнінг;
- зростаючий DAG-файл;
- широка інфраструктура Web3;
- простота перевірки транзакцій;
- доступність для майнерів без ASIC.

Зрівняння Ethereum Classic та Ethereum у таблиці 2.5.

Таблиця 2.5 – Аспекти Ethereum Classic та Ethereum

Аспект	Ethereum Classic	Ethereum (ETH)
Ідеологія	Незмінність, "code is law"	Гнучкість, зворотність змін
Консенсус	Proof of Work (Etchash)	Proof of Stake (Beacon Chain)
Розвиток	Консервативний, стабільний	Активне оновлення, EIP
Максимальна емісія	Так, 210 млн ETC	Ні, ETH інфляційний
Рівень розвитку DApps	Низький	Дуже високий
Основне призначення	Збереження цінностей, мінімалістичні DApp	Широка інфраструктура Web3

Хоча Ethereum Classic не має такої великої екосистеми, як Ethereum, платформа залишається активною. На базі ETC працюють:

- TokenMint – платформа створення токенів;
- Saturn Network – децентралізована біржа;
- Emerald Wallet – офіційний гаманець.

Криптовалюта ETC зберігається на популярних біржах (Binance, Kraken, Coinbase), підтримується гаманцями типу Ledger, MetaMask, Trust Wallet.

Ethereum Classic кілька разів зазнавав атаки типу "51%" – найвідоміші випадки були у 2019 та 2020 роках. Причина – низький хешрейт мережі, що дозволяло недобросовісним учасникам тимчасово захоплювати контроль над мережею. Це викликало занепокоєння, однак з часом безпеку було посилено завдяки новому механізму "MESS" (Modified Exponential Subjective Scoring), який ускладнює атаки на фінальні блоки. Структура Ethereum Classic зображена на рисунку 2.4.

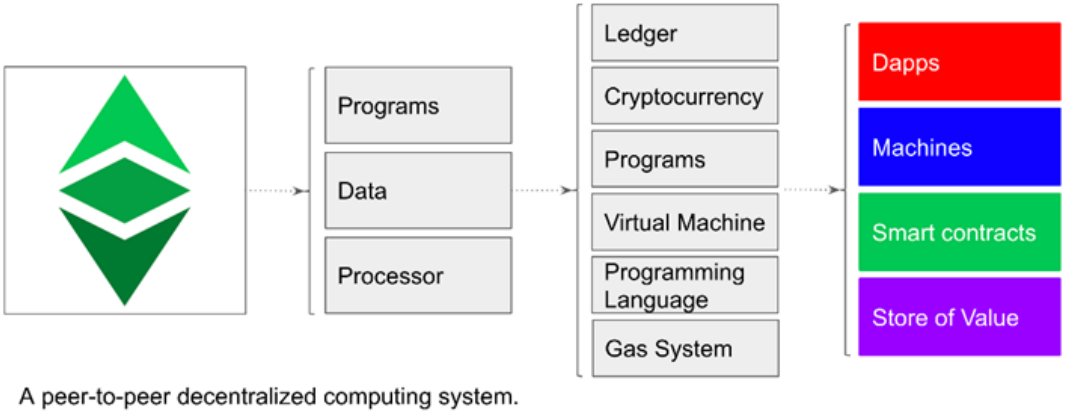

Рисунок 2.4 – Структура Ethereum Classic

Ethereum Classic є альтернативною реалізацією ідеї децентралізації, яка зберігає історичну спадщину Ethereum до хардфорку. Хоча він менш популярний, ніж Ethereum, ETC приваблює прихильників незмінності, PoW-майнерів і користувачів, які цінують передбачуваність емісії. У майбутньому Ethereum Classic може залишитися як енергозалежний "цифровий аналог золота", тоді як Ethereum – як є гнучкою багатофункціональною платформою Web3.

2.5 Ethereum Name Service

Ethereum Name Service (ENS) – це децентралізована система доменних імен, яка працює на основі блокчейну Ethereum. ENS надає можливість

перетворювати довгі та складні криптоадреси (напр., 0xAB12...f9D3) у зручні й читабельні імена, наприклад: ivanov.eth, university-cert.eth або student2025.univ.eth.

ENS є аналогом DNS (Domain Name System) у традиційному інтернеті, але функціонує на основі смарт-контрактів у блокчейні, що забезпечує децентралізацію, незмінність та відсутність контролю з боку централізованих реєстраторів.

Система доменних імен Ethereum Name Service (ENS) ґрунтується на двох ключових компонентах. Перший – це Registry (реєстр ENS), центральний смарт-контракт, що акумулює дані про власників доменів, їхні резолвери та час життя записів (TTL). Другий компонент – Resolver (резолвер ENS), контракт, призначений для конвертації імен у відповідні дані, такі як Ethereum-адреси, IPFS-хеші, публічні PGP-ключі чи описи профілів. ENS надає функціональність для прив'язки Ethereum-адрес до зручних для сприйняття імен, таких як ivanov.eth. Також, система дозволяє відображати NFT-активи та ENS-профілі в різних криптовалютних гаманцях (зокрема, MetaMask, Rainbow, Trust). Користувачі можуть створювати субдомени, наприклад, diploma.ivanov.eth, cert.kpi.eth або alice.univ.eth. Крім того, ENS дає змогу вказувати IPFS-ресурси, такі як сертифікати або сторінки студентів на IPFS, а також інтегрувати ENS з Web3-профілями за допомогою стандарту EIP-4361 (Sign-In With Ethereum), структуру якого можемо побачити на рисунку 2.5.

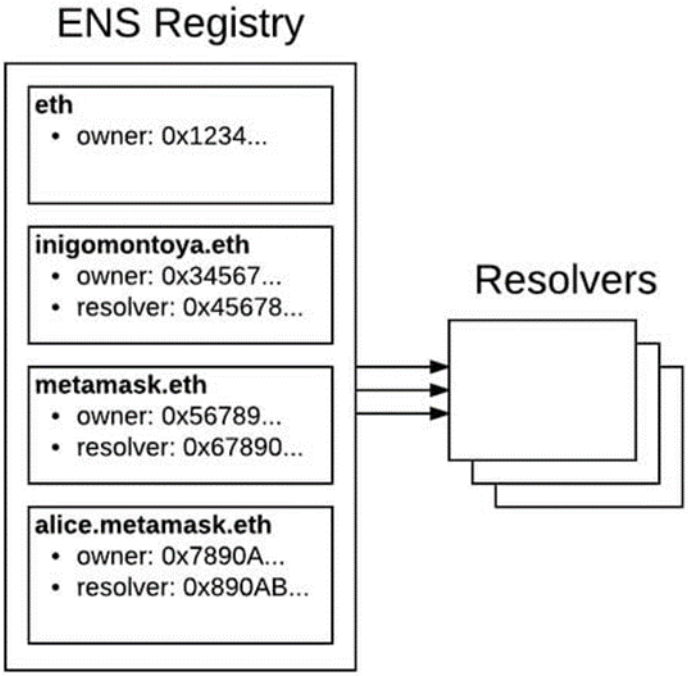

Рисунок 2.5 – Структура Ethereum Name Service

ENS має потенціал стати новим стандартом цифрової ідентичності для студентів, викладачів, університетів та освітніх програм. ENS надає можливості, такі як створення цифрового паспорта студента, де, наприклад, student2025.univ.eth може бути прив'язаний до профілю, NFT-дипломів, контактної інформації та навчальних досягнень. Також, ENS дозволяє використовувати освітні домени, такі як kpi.eth, khnu.edu.eth, або certificates.unesco.eth, які є офіційними ENS-доменами освітніх закладів. Можлива видача NFT-сертифікатів з ENS-підписом, що підтверджує справжність джерела (наприклад, courses.nure.eth). Роботодавці можуть автоматично перевіряти дипломи, скануючи ENS або пов'язаний IPFS-хеш для отримання хешу документа або підтвердження його автентичності.

Переваги ENS включають децентралізований контроль, що дозволяє користувачам повністю контролювати свій ENS-домен без залежності від реєстратора. Записи в ENS є прозорими, оскільки всі вони доступні в блокчейні Ethereum. ENS-домен також забезпечує швидку верифікацію походження, дозволяючи легко перевірити автора або власника ресурсу. Крім

того, ENS сумісний з Web3-технологіями, інтегруючись з MetaMask, IPFS, DeFi та NFT-маркетплейсами.

ENS активно розвивається з 2017 року, за підтримки Ethereum Foundation, OpenEthereum, Chainlink Labs та інших структур Web3. У 2021 році ENS здійснив дистрибуцію токенів ENS DAO, що стало першим прецедентом децентралізованого управління доменною системою. ENS інтегрований з такими платформами як WalletConnect, Etherscan, Uniswap, Arweave, Brave Browser та Lens Protocol.

Отже, ENS є не просто системою доменних імен, а ключовим елементом Web3-ідентичності, що здатний значно спростити зберігання та перевірку освітніх даних у блокчейні. У контексті токенізації освіти, ENS може стати основою для створення персоналізованого, відкритого та контрольованого студентом портфоліо в децентралізованому середовищі.

2.6 Self-Sovereign Identity

Концепція Self-Sovereign Identity (SSI) представляє децентралізовану цифрову ідентичність, де індивід самостійно володіє, контролює та управляє своїми персональними даними без необхідності залучення централізованих посередників, таких як державні реєстри або соціальні мережі. Основна ідея SSI полягає в наданні кожній особі можливості створювати, зберігати та використовувати власні ідентифікаційні дані через криптографічно захищену систему, що базується на технології блокчейну.

Як показано на рисунку 2.6, ключовими складовими SSI є децентралізовані ідентифікатори (DID) – унікальні глобальні ідентифікатори (наприклад, did:ethr:0xabc123...), що функціонують незалежно від централізованих органів. DID пов'язуються з парами публічного та приватного ключів, які забезпечують автентифікацію користувача. Інший важливий елемент – це Verifiable Credentials (VC), структуровані цифрові сертифікати або атестати (такі як дипломи, довідки чи сертифікати курсів),

які можуть бути незалежно перевірені та підписані освітньою установою. Третім компонентом є ідентифікаційний гаманець (Wallet SSI), що є додатком для зберігання DID, приватних ключів та набору підтверджень (VC), надаючи користувачеві повний контроль над даними та можливість їх поширення за потребою.

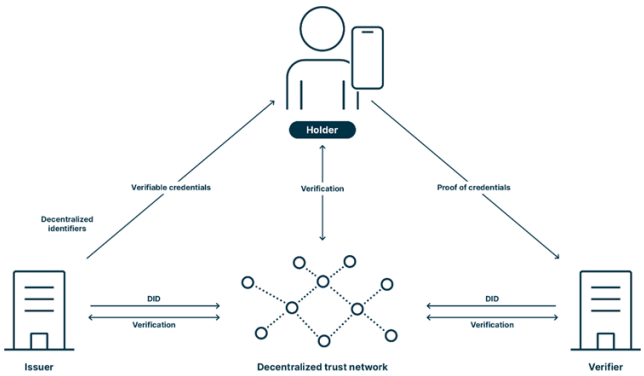

Рисунок 2.6 – Структура SSI

Наприклад, студент може отримати Verifiable Credential у вигляді токенізованого диплома або сертифіката, підписаного DID-ідентифікатором університету. Цей документ зберігається в ідентифікаційному гаманці студента. При працевлаштуванні студент може підтвердити володіння дипломом за допомогою свого DID, уникнувши надсилання копій, звернень до університету або ризику підробок.

Переваги SSI охоплюють повний контроль користувача над даними, оскільки вони зберігаються лише у власника, який вирішує, що і коли показувати. Дипломи та сертифікати можна перевіряти в автоматичному режимі, що забезпечує високу перевірюваність. Обмін даними здійснюється з дотриманням конфіденційності, без передачі всієї інформації. SSI підтримує інтероперабельність з різними платформами, зокрема Ethereum, Hyperledger та Solana. Крім того, технологія мінімізує потребу в посередниках, зменшуючи залежність від довірених третіх сторін.

Технологічні платформи, що підтримують SSI, включають стандарт Ethereum + DID:ethr для ідентифікації в мережі Ethereum, open-source-платформу Hyperledger Indy, модульну систему децентралізованих профілів Ceramic Network + IDX, а також готові рішення для цифрової ідентифікації, такі як Dock, Bloom та BrightID.

Серед реальних прикладів впровадження SSI можна назвати EBSI (European Blockchain Services Infrastructure), де ЄС розробляє систему децентралізованої ідентифікації для громадян і студентів у рамках програми Erasmus+, що стане основою цифрових дипломів. Массачусетський технологічний інститут (MIT) видає цифрові дипломи на блокчейні з DID, доступні в додатку Blockcerts.

Порівнюючи централізовану ідентичність та SSI, можна виділити ключові відмінності. У централізованій системі власність даних належить установі або державному органу, тоді як у SSI – користувачеві. Передача даних відбувається через запити або копії у централізованій моделі, а в SSI – через криптографічний доказ. Перевірка у централізованих системах здійснюється вручну або через API, тоді як SSI забезпечує автоматичну та миттєву перевірку. Безпека централізованої ідентичності залежить від бази даних, тоді як SSI базується на криптографії та блокчейні. Стосовно приватності, у централізованій системі вона часто порушується, тоді як користувач SSI сам обирає, що показувати.

Таким чином, Self-Sovereign Identity є фундаментальною технологією для побудови приватної, перевірюваної та децентралізованої цифрової ідентичності. У сфері освіти SSI може забезпечити персоналізовані, захищені та мобільні профілі студентів, що будуть корисними для навчання, кар'єрного зростання та на міжнародному рівні. У поєднанні з ENS, IPFS та смарт-контрактами, SSI формує нову модель освітньої екосистеми, що повністю належить студентам.

2.7 DAO

Децентралізована автономна організація (DAO) – це інноваційна модель управління, що функціонує на базі смарт-контрактів і не має єдиного централізованого керівництва. DAO дозволяє людям з усього світу об'єднуватися для прийняття спільних рішень та управління ресурсами через блокчейн, усуваючи потребу в традиційних ієрархічних структурах, таких як директори, офіси чи бюрократія. Це нова парадигма управління в епоху Web3, де логіка організації чітко зафіксована в програмному коді смарт-контрактів, підкреслюючи принцип "код – це закон".

Основні принципи функціонування DAO включають автономність, де управління повністю здійснюється смарт-контрактами без ручного втручання. Прозорість забезпечується тим, що всі правила, бюджети, результати голосувань та прийняті рішення доступні в блокчейні. Децентралізоване голосування гарантує, що кожен учасник має право голосу, пропорційне його токенам або ролі в організації. Смарт-контракти автоматизують виконання рішень, таких як виділення коштів або зміна параметрів. Участь у DAO є відкритою, дозволяючи будь-кому приєднатися до спільноти.

Більшість DAO реалізовані на платформі Ethereum, використовуючи ряд спеціалізованих інструментів. Смарт-контракти управляють казначейством, правами доступу та процесами голосування. Токени управління (Governance Tokens) визначають вплив користувача в системі. Для проведення голосувань застосовуються спеціалізовані протоколи, такі як Snapshot, Tally та Aragon DAO. Доступ до участі забезпечується через інтерфейси, зокрема DAOhaus, Boardroom та Juicebox.

У сфері освіти DAO можуть стати новою моделлю для управління освітніми проєктами, фондами та спільнотами. Наприклад, в моделі DAO-факультету студенти, викладачі та випускники можуть голосувати за розподіл бюджету, створення нових курсів або надання стипендій. DAO-

грантова програма дозволяє випускникам та спонсорам спільно визначати отримувачів стипендій або грантів. Концепція DAO-університету передбачає децентралізовану організацію, де учасники самостійно розробляють курси, оцінюють їхню цінність та сертифікують результати. Також, в рамках Peer review DAO, оцінювання дослідницьких робіт може відбуватися шляхом децентралізованого голосування експертів.

Переваги DAO включають прозоре прийняття рішень, оскільки кожен учасник може бачити, хто і як голосує. Децентралізоване казначейство забезпечує автоматичний розподіл коштів. DAO сприяє інклюзивності, дозволяючи студентам впливати на університетські процеси. Автоматизація процесів, таких як виділення стипендій або перевірка результатів, відбувається без адміністративного втручання. Крім того, DAO є глобальними і не мають географічних чи юридичних обмежень, структуру DAO можемо побачити на рисунку 2.7.

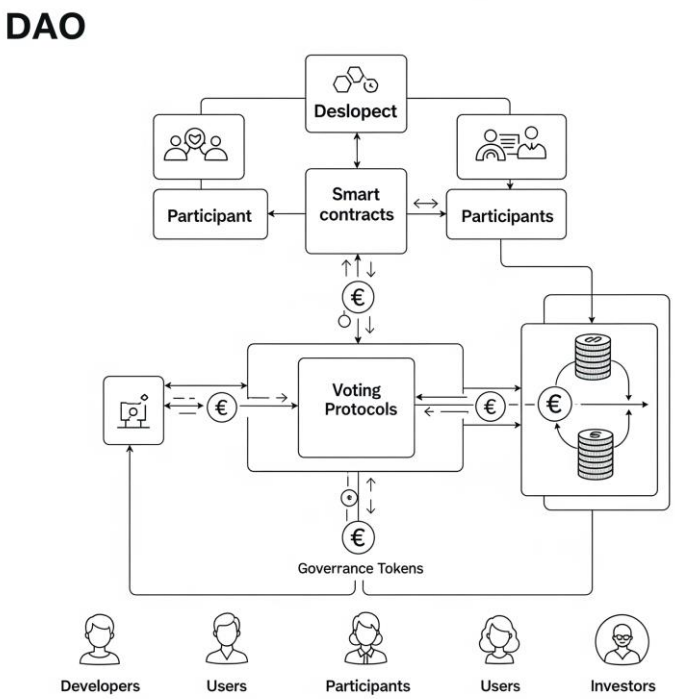

Рисунок 2.7 – Структура DAO

Однак, існують і певні недоліки та виклики. Безпека смарт-контрактів залишається критичною, оскільки помилка в коді може призвести до втрати коштів, як це сталося з The DAO у 2016 році. Проблемою може бути низька участь, коли багато учасників не беруть участь у голосуваннях. Також існує ризик концентрації влади, якщо великі власники токенів контролюють голосування. Правова невизначеність є ще одним викликом, оскільки DAO не завжди вписуються в існуючі національні правові системи.

DAO ідеально інтегруються з концепцією токенізованої освіти. DAO можуть видавати NFT-сертифікати , керувати платформами токенізованих курсів , прозоро розподіляти навчальні гранти , а також мати власний токен для стимулювання участі або валідації знань.

Підсумовуючи, DAO є перспективною формою колективної діяльності, яка має особливе значення для майбутнього відкритої, децентралізованої та токенізованої освіти. DAO, побудовані на блокчейні Ethereum, можуть забезпечити прозоре та ефективне управління освітніми проєктами, розподіл ресурсів, формування навчальних програм та підтримку академічної мобільності.

2.8 Zero-Knowledge Proofs (ZKPs)

Докази з нульовим розголошенням (Zero-Knowledge Proofs, ZKP) – це криптографічний метод, який дозволяє одній стороні (доказувачу) підтвердити іншій стороні (перевіряючому) достовірність певного твердження, не розкриваючи при цьому саму інформацію, принцип роботи можна візуалізувати на рисунку 2.8. Іншими словами, доказувач може продемонструвати знання певної інформації (наприклад, пароля, результату або навички), не розкриваючи її вміст. Уявіть ситуацію, коли студент хоче довести отримання диплома з певного університету, не надаючи сам PDF-документ, його номер чи навіть назву навчального закладу. Завдяки ZKP можна створити підтвердження володіння справжнім дипломом, яке можна

миттєво перевірити без безпосереднього доступу до документа.

Zero-Knowledge Proof мають три ключові властивості. Повнота (Completeness) означає, що якщо твердження істинне, чесний перевіряючий завжди отримає підтвердження. Коректність (Soundness) гарантує, що якщо твердження хибне, шахрай не зможе переконати перевіряючого у його правдивості. Нульове розголошення (Zero-Knowledge) є основною властивістю, що забезпечує конфіденційність: перевіряючий не дізнається нічого про саму інформацію, окрім факту її коректності.

Існують різні типи ZKP. Інтерактивні докази з нульовим розголошенням (Interactive Zero-Knowledge Proofs) вимагають взаємодії між доказувачем та перевіряючим, наприклад, у багатоетапному підтвердженні знання пароля. Натомість, неінтерактивні докази з нульовим розголошенням (Non-Interactive Zero-Knowledge Proofs, NIZK) створюються один раз і можуть бути перевірені без повторної участі доказувача; цей тип широко застосовується в блокчейні, зокрема zk-SNARKs та zk-STARKs.

Серед NIZK особливо виділяються zk-SNARKs (Succinct Non-interactive Argument of Knowledge), що характеризуються малим об'ємом доказу та швидкою перевіркою, і використовуються в таких системах, як zkSync та Zcash . zk-STARKs мають вищий рівень безпеки, не потребуючи "довіреної установки" (trusted setup), та придатні для великих обчислень, знаходячи застосування, наприклад, у StarkNet.

Сценарії застосування ZKP в освіті є різноманітними. Наприклад, для доказу диплома без розголошення, кандидат може підтвердити наявність диплома за допомогою ZKP, не надаючи сам документ або назву університету. Це дозволяє студентам брати участь в анонімному голосуванні в DAO, наприклад, при виборі факультативів, не розкриваючи свою особистість, але підтверджуючи право голосу. ZKP також забезпечують захист навчального прогресу, дозволяючи користувачеві підтвердити завершення курсу або успішне складання тесту без публікації оцінок чи інших деталей. Крім того, вони можуть використовуватися для подання

заявок на стипендії з конфіденційністю, надаючи доказ відповідності вимогам (бал, статус) без розкриття чутливих деталей.

Переваги ZKP включають забезпечення конфіденційності, оскільки чутливі дані не потрібно розкривати. Швидка перевірка вимагає мінімальних затрат часу та обчислень. Технологія також сприяє мінімізації даних у блокчейні, зменшуючи кількість записів та витрати. Гнучкість у дизайні децентралізованих додатків (DApps) дозволяє створювати приватні освітні застосунки. Нарешті, ZKP забезпечує міжнародну верифікацію дипломів без необхідності розкриття повного документа.

Таким чином, Zero-Knowledge Proofs є ключовою технологією для інтеграції приватності та перевірюваності в блокчейн-освіту. Вони дозволяють студентам, викладачам, роботодавцям та навчальним установам прозоро взаємодіяти, зберігаючи при цьому конфіденційність. ZKP відкривають шлях до створення цифрових дипломів, тестів, систем голосування та DAO, де довіра забезпечується не людським фактором, а математичними алгоритмами

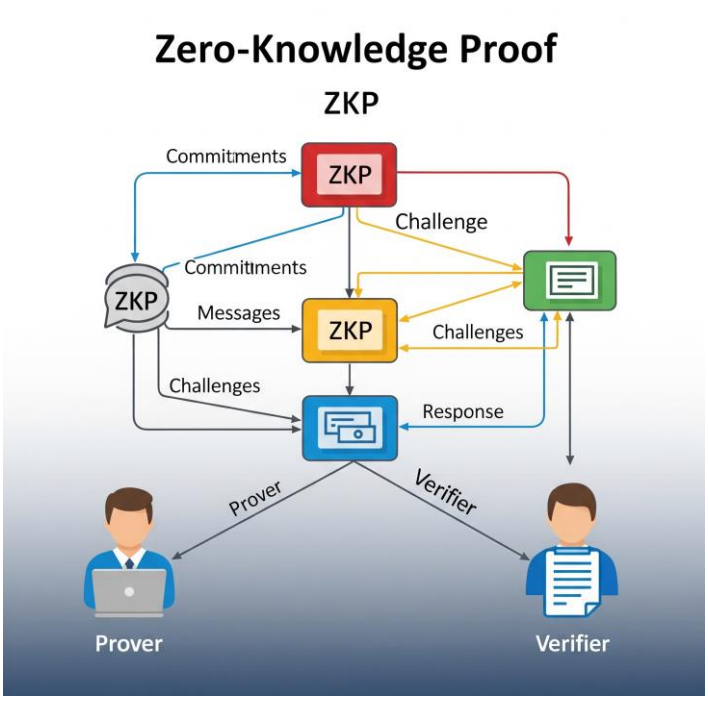

Рисунок 2.8 – Принцип роботи Zero-Knowledge Proof (ZKP)

2.9 Підсумки до другого розділу

У другому розділі було проведено всебічний аналіз ключових технологічних складових, що формують фундамент токенізації освітніх активів у децентралізованому просторі. Було розглянуто поняття блокчейну та його види, включаючи публічні, приватні, консорціумні та гібридні блокчейни. Детально проаналізовано роль хешування у блокчейні, зокрема криптографічні хеш-функції SHA-256 та Ethash, що забезпечують цілісність та безпеку даних.

Особлива увага приділялася платформі Ethereum, її віртуальній машині (EVM), смарт-контрактам та стандартам токенів (ERC-20, ERC-721, ERC-1155). Розглянуто перехід Ethereum до Proof of Stake та роль Layer 2 рішень у масштабуванні мережі. Також було проведено порівняльний аналіз з Ethereum Classic, висвітлюючи його ідеологію незмінності та використання Proof of Work.

Досліджено роль Ethereum Name Service як децентралізованої системи доменних імен та її потенціал у цифровій ідентифікації студентів та освітніх установ. Проаналізовано концепцію Self-Sovereign Identity та її компоненти, такі як Decentralized Identifiers (DID) та Verifiable Credentials, що дозволяють користувачам контролювати свої особисті дані. Окремо розглянуто децентралізовані автономні організації (DAO) як нову модель управління в освіті. Важливим аспектом стало вивчення Zero-Knowledge Proofs (ZKPs), що забезпечують конфіденційність даних без розкриття самої інформації.

Таким чином, у результаті дослідження було підтверджено, що сучасні технології блокчейн, у поєднанні з криптографічними протоколами, децентралізованими сервісами та Web3-підходами, забезпечують надійний фундамент для побудови нової моделі збереження, верифікації та контролю освітніх активів. Ці технології створюють передумови для підвищення прозорості, автономії та довіри в освітньому процесі.

3 ПРАКТИЧНА ЧАСТИНА. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура системи

Розроблене програмне забезпечення функціонує як децентралізований застосунок (dApp), що інтегрує три основні компоненти для забезпечення токенізації освітніх активів та взаємодії з блокчейн-мережею Ethereum. Загальна архітектура системи представлена на рисунку 3.1.

Першим і центральним компонентом є смарт-контракт, розроблений мовою Solidity. Цей контракт відповідає за безпосереднє зберігання освітніх повідомлень у блокчейні Ethereum, забезпечуючи їх незмінність та прозорість. Він містить структури даних для повідомлень та методи для їх запису, читання та підрахунку.

Другим ключовим елементом є клієнтська частина (frontend), реалізована за допомогою React.js. Ця частина відповідає за відображення інтерфейсу користувача, дозволяючи йому взаємодіяти з системою. Інтерфейс включає функціонал для підключення гаманця MetaMask, відображення балансу, введення та надсилання текстових повідомлень, виведення списку збережених повідомлень, перемикання акаунтів та зміни теми інтерфейсу. Підключення розширення можемо побачити на рисунку 3.2.

Третій компонент – бібліотека Web3.js, яка слугує сполучною ланкою, забезпечуючи взаємодію між смарт-контрактом і фронтендом через Ethereum-провайдер, такий як MetaMask. Web3.js дозволяє клієнтській частині надсилати транзакції, викликати функції смарт-контракту та отримувати дані з блокчейну.

Таким чином, архітектура системи забезпечує повний цикл взаємодії користувача з блокчейном: від ініціації транзакції через зручний інтерфейс до її обробки смарт-контрактом та відображення результатів. Цей підхід створює децентралізоване рішення для токенізації освітніх активів, де дані

захищені блокчейном, а користувач має прямий контроль над своїми записами через криптографічний гаманець.

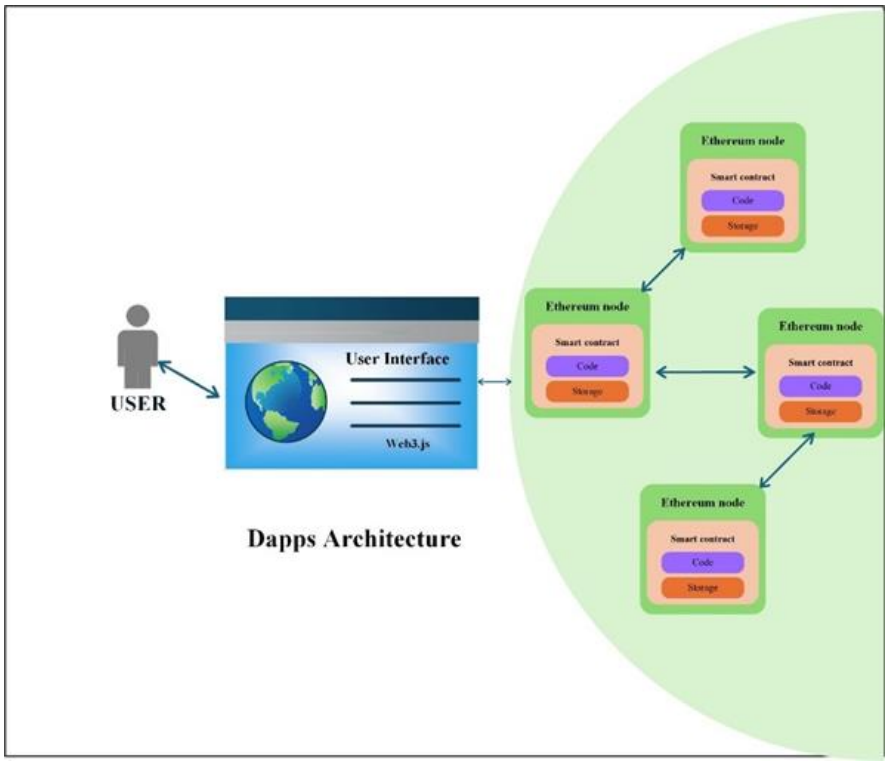

Рисунок 3.1 – Загальна архітектура системи

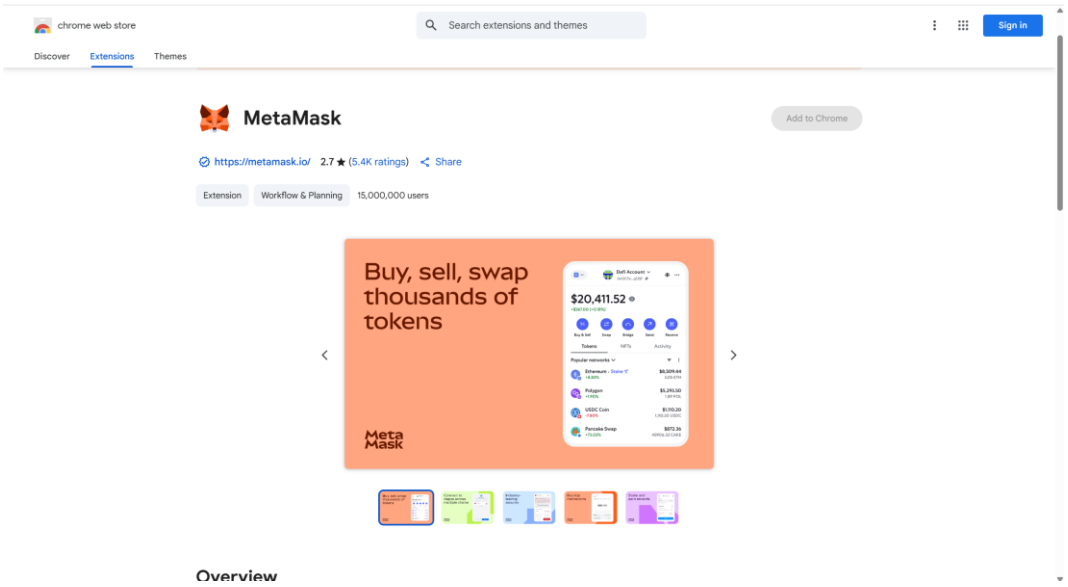

Рисунок 3.2 – Розширення Metamask

3.2 Смарт-контракт

Серцем розробленої системи є смарт-контракт, реалізований мовою Solidity (версія 0.8.12). Цей контракт виконує ключову роль у токенізації освітніх активів, забезпечуючи їх децентралізоване та незмінне зберігання в блокчейні Ethereum. Він функціонує як основний механізм для взаємодії з даними, надаючи можливість записувати та отримувати освітні повідомлення.

Контракт Storage реалізує структуру даних Message, яка спроєктована для ефективного зберігання інформації. Кожен об'єкт Message містить три основні поля: text, що представляє зміст освітнього повідомлення; createdAt – ціле число типу uint256, що фіксує мітку часу створення повідомлення; та sender – адреса облікового запису Ethereum (address), який відправив повідомлення.

Для управління цими даними смарт-контракт надає функцію set(string _text), призначену для запису нового повідомлення до блокчейну. Вона дозволяє користувачам додавати освітні активи у вигляді текстових записів. Перед збереженням функція виконує перевірку, чи не є повідомлення порожнім або занадто довгим, та чи є відправник коректною адресою. У разі успішного додавання генерується подія MessageStored, яка містить адресу відправника, текст та мітку часу, що полегшує відстеження змін у блокчейні зовнішніми застосунками. Також контракт надає функцію getMessages(), що доступна для читання (view) та повертає всі збережені повідомлення у вигляді масиву Message[]. Ще одна функція для читання, getMessageCount(), повертає загальну кількість збережених повідомлень у сховищі.

Крім того, контракт включає визначення трьох кастомних помилок: EmptyMessage(), MessageTooLong() та InvalidSender(). Ці помилки допомагають забезпечити цілісність даних та краще керувати відмовами при спробах записати некоректні повідомлення. Максимальна дозволена довжина повідомлення MAX_MESSAGE_LENGTH встановлена на рівні 280 символів.

Взаємодія зі смарт-контрактом відбувається через транзакції, які підписуються користувачами за допомогою їхніх криптографічних гаманців (наприклад, MetaMask). Кожна транзакція, що змінює стан блокчейну (наприклад, виклик функції set), вимагає оплати комісії (gas), що стимулює валідаторів мережі обробляти її. Детальний лістинг смарт-контракту наведено в Лістингу 3.1.

Цей смарт-контракт є надійним фундаментом для токенізації освітніх даних, оскільки він забезпечує незмінність записів, прозорість доступу та можливість програмної взаємодії з освітніми активами через блокчейн

Лістинг 3.1 – Реалізація смарт-контракту (файл Storage.sol)

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.12;
contract Storage {
    struct Message {
        string text;
        uint256 createdAt;
        address sender;
    }
    Message[] private store;
    error EmptyMessage();
    error MessageTooLong();
    error InvalidSender();

    event MessageStored(
        address indexed sender,
        string text,
        uint256 timestamp
    );
    uint256 public constant MAX_MESSAGE_LENGTH = 280;
    function set(string calldata _text) external {
        if (msg.sender == address(0)) revert InvalidSender();
        uint256 textLength = bytes(_text).length;
        if (textLength == 0) revert EmptyMessage();
        if (textLength > MAX_MESSAGE_LENGTH) revert
MessageTooLong();
        store.push(Message({
            text: _text,
            createdAt: block.timestamp,
            sender: msg.sender
        }));
        emit MessageStored(msg.sender, _text, block.timestamp);
    }
    function getMessages() external view returns (Message[]
memory) {
```

```
        return store;
    }
    function getMessageCount() external view returns (uint256) {
        return store.length;
    }
}
```

3.3 Налаштування середовища розробки

Для Ефективна розробка та розгортання смарт-контрактів вимагають ретельно налаштованого середовища. Для забезпечення зручності компіляції, міграції та тестування смарт-контракту в цьому проєкті використовується фреймворк Truffle.

Truffle – це один з найпопулярніших фреймворків для розробки додатків на базі Ethereum Virtual Machine (EVM). Він надає розробникам повний набір інструментів, необхідних для всього життєвого циклу децентралізованого застосунку (dApp).

Основні можливості Truffle включають автоматизацію процесу компіляції файлів Solidity у байт-код та ABI (Application Binary Interface), які необхідні для взаємодії зі смарт-контрактами в блокчейні. Фреймворк також дозволяє легко розгортати скомпільовані смарт-контракти у різні блокчейн-мережі – будь то локальна тестова мережа (наприклад, Ganache), публічні тестові мережі (Ropsten, Goerli) або основна мережа Ethereum. Це забезпечується механізмом міграцій, що дозволяє контролювати порядок розгортання контрактів та їхні залежності.

Крім того, Truffle надає вбудовану тестову інфраструктуру, яка дозволяє писати автоматизовані тести для смарт-контрактів мовами JavaScript або Solidity, що є критично важливим для забезпечення надійності та безпеки коду смарт-контрактів. Також Truffle пропонує стандартизовану структуру проєкту, що полегшує організацію файлів, бібліотек та залежностей.

У даному проєкті конфігурація Truffle, деталі якої наведені в Лістингу 3.2, налаштована для роботи в локальному середовищі розробки. Це

забезпечує швидке тестування та налагодження без необхідності взаємодії з реальними публічними мережами, що значно прискорює процес розробки.

Зокрема, у конфігурації вказано, що мережа development буде працювати на хості 127.0.0.1 та порту 8545. Параметр network_id: "*" дозволяє Truffle підключатися до будь-якої мережі, яка запущена на цьому хості та порту, що є зручним для використання з Ganache. Для компілятора Solidity (solc) встановлена версія 0.8.12 та активований оптимізатор (optimizer) з 200 запусками, що сприяє зменшенню розміру розгорнутого контракту та оптимізації витрат газу, що сприяє зменшенню плати за кожну транзакцію.

Таким чином, використання Truffle значно спрощує процеси розробки та тестування, надаючи комплексний інструментарій для взаємодії зі смарт-контрактами та блокчейном.

Лістинг 3.2 - Конфігурація Truffle (файл truffle-config.js)

```
module.exports = {
  networks: {
    development: {
      host: "127.0.0.1",
      port: 8545,
      network_id: "*",
    }
  },
  compilers: {
    solc: {
      version: "0.8.12",
      settings: {
        optimizer: { enabled: true, runs: 200 }
      }
    }
  }
};
```

3.4 Реалізація клієнтської частини (React.js)

Клієнтська частина програмного забезпечення є критично важливим компонентом, що забезпечує взаємодію користувача з децентралізованою

системою. Вона реалізована за допомогою бібліотеки React.js, що дозволяє створювати динамічні та адаптивні інтерфейси. Для забезпечення повноцінного функціоналу були інтегровані ключові інструменти : Web3.js, яка служить основним мостом для підключення до блокчейну Ethereum, дозволяючи клієнтській частині взаємодіяти зі смарт-контрактами, надсилати транзакції та отримувати дані з мережі Ethereum. Розширення для браузера MetaMask використовується для авторизації користувачів та підпису транзакцій, забезпечуючи безпечне управління обліковими записами Ethereum та їхніми балансами, а також дозволяючи користувачам підтверджувати транзакції перед їх відправленням у мережу. Для стилізації інтерфейсу використано бібліотеку styled-components, яка дозволяє створювати компоненти з вбудованими стилями, що підвищує модульність та спрощує управління зовнішнім виглядом застосунку. Бібліотека framer-motion застосована для реалізації анімацій елементів інтерфейсу, роблячи взаємодію з застосунком більш плавною та візуально привабливою. Для збереження обраної користувачем теми (світлої/темної) використовується localStorage браузера, що дозволяє зберігати налаштування користувача між сесіями.

Основні функції інтерфейсу включають підключення гаманця MetaMask, відображення балансу користувача, введення текстового повідомлення та його надсилання до блокчейну, вивід списку повідомлень, отриманих з мережі, а також можливість перемикання акаунтів та відключення гаманця. Додаток також підтримує перемикання теми інтерфейсу між світлим та темним режимами.

Далі представлено розділений на частини код файлу App.js з поясненнями до кожної важливої функції.

Лістинг 3.3 – Імпорти та визначення тем

```
import './App.css';
import Web3 from 'web3';
import Storage from './contracts/Storage.json';
```

```
import { useEffect, useRef, useState } from 'react';
import styled, { ThemeProvider } from 'styled-components';
import { motion, AnimatePresence } from 'framer-motion';

const lightTheme = {
  background: '#f2f2f2',
  text: '#000',
  card: '#fff',
  button: '#007bff',
  buttonText: '#fff',
};
const darkTheme = {
  background: '#121212',
  text: '#fff',
  card: '#1e1e1e',
  button: '#1e88e5',
  buttonText: '#fff',
};
```

Цей блок імпортує необхідні бібліотеки та компоненти: Web3 для взаємодії з Ethereum, Storage.json – ABI та адреси розгорнутого смарт-контракту, хуки React (useEffect, useRef, useState), styled-components для стилізації та framer-motion для анімацій. Також визначаються дві теми (lightTheme та darkTheme) для візуального оформлення застосунку.

Лістинг 3.4 – Стилізовані компоненти React

```
const AppWrapper = styled.div`
  background-color: ${(props) => props.theme.background};
  color: ${(props) => props.theme.text};
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
`;

const Card = styled.div`
  background-color: ${(props) => props.theme.card};
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
  padding: 2rem;
  border-radius: 10px;
  width: 90%;
  max-width: 500px;
`;
const Button = styled.button`
  background-color: ${(props) => props.theme.button};
  color: ${(props) => props.theme.buttonText};
```

```
  border: none;
  padding: 10px 20px;
  border-radius: 6px;
  cursor: pointer;
  margin-top: 1rem;
`;

const WalletButton = styled.div`
  position: absolute;
  top: 20px;
  right: 20px;
`;
const ThemeToggler = styled.div`
  position: absolute;
  bottom: 20px;
  right: 20px;
`;

const LogoBlock = styled.div`
  position: absolute;
  top: 20px;
  left: 20px;
  display: flex;
  align-items: center;
  gap: 10px;
  font-weight: bold;
  font-size: 18px;
`;
const LogoImage = styled.img`
  width: 28px;
  height: 28px;
`;
```

Цей розділ визначає стилізовані React-компоненти за допомогою styled-components. Кожен компонент (AppWrapper, Card, Button, WalletButton, ThemeToggler, LogoBlock, LogoImage) отримує стилі, які динамічно змінюються залежно від обраної теми, забезпечуючи адаптивний та привабливий дизайн інтерфейсу.

Лістинг 3.5 – Ініціалізація стану та функція initWeb3

```
function App() {
  const [web3, setWeb3] = useState(null);
  const [account, setAccount] = useState(null);
  const [allAccounts, setAllAccounts] = useState([]);
  const [balance, setBalance] = useState("0");
  const [text, setText] = useState("");
  const [messages, setMessages] = useState([]);
  const [theme, setTheme] = useState(() =>
```

```
localStorage.getItem('theme') || 'light');
  const [isWalletMenuOpen, setIsWalletMenuOpen] =
useState(false);
  const walletMenuRef = useRef(null);
  const isDark = theme === 'dark';

  const toggleTheme = () => {
    const nextTheme = isDark ? 'light' : 'dark';
    setTheme(nextTheme);
    localStorage.setItem('theme', nextTheme);
  };

  const initWeb3 = async () => {
    if (window.ethereum) {
      try {
        await window.ethereum.request({ method:
'eth_requestAccounts' });
        const web3Instance = new Web3(window.ethereum);
        setWeb3(web3Instance);
        return web3Instance;
      } catch (err) {
        console.error("Access denied to MetaMask:", err);
      }
    } else {
      alert("Please install MetaMask.");
    }
  };
```

У цьому блоці ініціалізується стан компонентів за допомогою хуків useState для управління даними (стан web3, поточний обліковий запис account, список усіх доступних облікових записів allAccounts, баланс balance, текст повідомлення text, список повідомлень messages, поточна тема theme, стан меню гаманця isWalletMenuOpen). useRef використовується для посилання на елемент меню гаманця. Функція toggleTheme перемикає світлу/темну тему та зберігає її в localStorage. Функція initWeb3 асинхронно ініціалізує об'єкт Web3 шляхом запиту доступу до облікових записів MetaMask, повертаючи екземпляр web3Instance або обробляючи помилки.

Лістинг 3.6 – Функції завантаження даних та підключення/відключення гаманця

```
const loadData = async (web3Instance, accountAddr) => {
    const id = await web3Instance.eth.net.getId();
    const network = Storage.networks[id];
    if (!network) {
```

```
      alert("Smart contract not deployed on this network.");
      return;
    }
    const contract = new web3Instance.eth.Contract(Storage.abi,
network.address);
    const balanceWei = await
web3Instance.eth.getBalance(accountAddr);
    const balanceEth = web3Instance.utils.fromWei(balanceWei,
'ether');

    const storedMessages = await
contract.methods.getMessages().call();

    setBalance(balanceEth);
    setMessages(storedMessages);
  };

  const handleConnect = async () => {
    const web3Instance = await initWeb3();
    if (!web3Instance) return;
    const accounts = await web3Instance.eth.getAccounts();
    if (accounts.length > 0) {
      setAccount(accounts[0]);
      setAllAccounts(accounts);
      await loadData(web3Instance, accounts[0]);
    }
  };

  const handleAccountSwitch = async (newAccount) => {
    setAccount(newAccount);
    await loadData(web3, newAccount);
    setIsWalletMenuOpen(false);
  };

  const handleDisconnect = () => {
    setAccount(null);
    setMessages([]);
    setBalance("0");
    setIsWalletMenuOpen(false);
  };
```

Функція loadData отримує ідентифікатор мережі, перевіряє розгортання контракту, ініціалізує екземпляр контракту та завантажує баланс поточного облікового запису та всі збережені повідомлення зі смарт-контракту. Функція handleConnect викликає initWeb3 для підключення до MetaMask, отримує список облікових записів та завантажує дані для першого облікового запису. handleAccountSwitch оновлює поточний обліковий запис та повторно завантажує дані для нього, закриваючи при цьому меню гаманця.

handleDisconnect скидає стан застосунку до початкового, відключаючи гаманець.

Лістинг 3.7 – Функція надсилання повідомлення

```
const handleSend = async () => {
    if (!web3 || !account || !text) return;
    try {
      const id = await web3.eth.net.getId();
      const network = Storage.networks[id];
      if (!network) throw new Error("Contract not deployed on
this network");
      const contract = new web3.eth.Contract(Storage.abi,
network.address);
      await contract.methods.set(text).send({
        from: account,
        gas: 300000,
        gasPrice: web3.utils.toWei('20', 'gwei'),
      });
      const updatedMessages = await
contract.methods.getMessages().call();
      setMessages(updatedMessages);
      setText('');
    } catch (error) {
      console.error("Transaction failed:", error.message);
      alert("Transaction failed");
    }
  };
```

Функція handleSend відповідає за надсилання текстового повідомлення до смарт-контракту. Вона перевіряє наявність підключеного гаманця та тексту повідомлення. Якщо всі перевірки пройдені, функція отримує ідентифікатор мережі, ініціалізує контракт та викликає метод set смарт-контракту. Важливою частиною є вказання from, gas та gasPrice для транзакції. Після успішного надсилання повідомлення, функція оновлює список повідомлень в інтерфейсі та очищає поле введення. Обробка помилок передбачена для випадків, коли транзакція не вдалася.

Лістинг 3.8 – Хук useEffect для закриття меню гаманця

```
useEffect(() => {
    const handleClickOutside = (event) => {
      if (walletMenuRef.current &&
!walletMenuRef.current.contains(event.target)) {
```

```
        setIsWalletMenuOpen(false);
      }
    };

    if (isWalletMenuOpen) {
      document.addEventListener('mousedown',
handleClickOutside);
    }

    return () => {
      document.removeEventListener('mousedown',
handleClickOutside);
    };
  }, [isWalletMenuOpen]);
```

Цей хук useEffect відповідає за закриття меню вибору гаманця, якщо користувач клікає за його межами. Він додає слухача подій mousedown до document при відкритті меню та видаляє його при закритті або при розмонтуванні компонента, запобігаючи витокам пам'яті.

Лістинг 3.9 – Структура інтерфейсу (JSX)

```
return (
    <ThemeProvider theme={isDark ? darkTheme : lightTheme}>
      <AppWrapper>
        {/* Блок логотипу */}
        <LogoBlock>
          <LogoImage src="/favicon.png" alt="Logo" />
          EduChain
        </LogoBlock>

        {/* Кнопка гаманця та меню вибору акаунту */}
        <WalletButton>
          {!account ? (
            <Button onClick={handleConnect}>Connect
Wallet</Button>
          ) : (
            <div style={{ position: 'relative' }}>
              <Button onClick={() => setIsWalletMenuOpen((prev)
=> !prev)}>
                {account.slice(0, 6)}...{account.slice(-4)}
              </Button>
              <AnimatePresence>
                {isWalletMenuOpen && (
                  <motion.div ref={walletMenuRef} /* ... стилі
та анімації ... */>
                    <strong>Accounts:</strong>
                    <ul>
                      {allAccounts.map((acc) => (
```

```
                          <li key={acc}>
                            <button onClick={() =>
handleAccountSwitch(acc)} /* ... стилі ... */>
                              {acc.slice(0, 6)}...{acc.slice(-4)}
                            </button>
                          </li>
                        ))}
                      </ul>
                      <hr />
                      <button onClick={handleDisconnect} /* ...
стилі ... */>
                        Disconnect
                      </button>
                    </motion.div>
                  )}
                </AnimatePresence>
              </div>
            )}
          </WalletButton>

          {/* Перемикач теми */}
          <ThemeToggler>
            <button onClick={toggleTheme} /* ... стилі ... */>
              <AnimatePresence mode="wait" initial={false}>
                <motion.span key={theme} /* ... анімації ... */>
                  {isDark ? '🌞' : '🌙'}
                </motion.span>
              </AnimatePresence>
            </button>
          </ThemeToggler>

          {/* Основна картка взаємодії */}
          <Card>
            {account && (
              <>
                <strong>{account}</strong>
                <p>Balance: {parseFloat(balance).toFixed(6)}
ETH</p>
              </>
            )}
            <input
              value={text}
              onChange={(e) => setText(e.target.value)}
              placeholder="Enter your message"
              style={{ width: '100%', padding: '10px', marginTop:
'1rem' }}
            />
            <Button onClick={handleSend}>SEND</Button>

            {account && messages.length > 0 && (
              <>
                <h4>Stored Messages:</h4>
                {messages
```

```
                .filter((msg) => msg.sender.toLowerCase() ===
account.toLowerCase())
                .map((msg, i) => (
                  <div key={i}>
                    <strong>{msg.sender}</strong>: {msg.text}{'
'}
                    <em>({new Date(Number(msg.createdAt) *
1000).toLocaleString()})</em>
                  </div>
                ))}
            </>
          )}
        </Card>
      </AppWrapper>
    </ThemeProvider>
  );
```

Це основний рендер-метод компонента App, який визначає структуру інтерфейсу користувача. Він включає: обгортку AppWrapper з динамічними стилями теми, логотип EduChain в LogoBlock, кнопки для підключення/відключення гаманця та перемикання облікових записів у WalletButton (з анімаціями framer-motion для меню), перемикач теми в ThemeToggler.

Головна Card містить інформацію про підключений обліковий запис (адреса та баланс), поле для введення повідомлень та кнопку "SEND". Також відображається відфільтрований список повідомлень, що належать поточному користувачеві, з часом створення.

3.5 Взаємодія з MetaMask

Інтеграція з гаманцем MetaMask є ключовим аспектом функціонування розробленого децентралізованого застосунку, оскільки саме він слугує основним інтерфейсом для взаємодії користувача з мережею Ethereum. MetaMask – це розширення для веб-браузера, яке діє як криптогаманець, надаючи безпечний спосіб керувати обліковими записами Ethereum, зберігати цифрові активи та підписувати транзакції.

Реалізація взаємодії з MetaMask здійснюється через бібліотеку Web3.js, що була детально розглянута в попередньому підрозділі. Ця бібліотека

дозволяє застосунку звертатися до провайдера window.ethereum, який інжектується MetaMask у середовище браузера. Після успішного підключення користувач отримує можливість переглядати баланс у мережі Ethereum , здійснювати транзакції з записом повідомлень , змінювати обліковий запис або відключати його. Зокрема, застосунок може запитувати та відображати поточний баланс ETH підключеного облікового запису користувача. Це реалізується за допомогою функції web3.eth.getBalance(accountAddr), яка повертає баланс у Wei (найменшій одиниці ETH), що потім конвертується у ETH для зручності відображення.

При здійсненні транзакцій з записом повідомлень, MetaMask виводить запит на підтвердження, де користувач може переглянути деталі (наприклад, вартість газу) та авторизувати операцію. Це забезпечує прозорість та безпеку, оскільки жодна транзакція не може бути виконана без явного схвалення власника гаманця. Застосунок також відстежує зміни активного облікового запису в MetaMask. Якщо користувач перемикає обліковий запис безпосередньо в розширенні MetaMask, інтерфейс застосунку автоматично оновлює відображувану адресу, баланс та фільтрує повідомлення, показуючи лише ті, що належать новому активному обліковому запису. Користувач може відключити свій гаманець від застосунку, що призводить до очищення локального стану даних облікового запису та повернення інтерфейсу до початкового стану, вимагаючи повторного підключення для подальшої взаємодії.

Важливо відзначити, що безпека взаємодії значною мірою покладається на MetaMask, який обробляє приватні ключі користувачів у безпечному середовищі, ізольованому від самого застосунку. Це мінімізує ризики компрометації облікових записів. Таким чином, MetaMask є не просто інструментом для транзакцій, а невід'ємною частиною архітектури dApp, що гарантує децентралізовану авторизацію та безпечну взаємодію з блокчейном.

3.6 UI/UX компоненти

Розроблений застосунок має інтуїтивно зрозумілий та функціональний інтерфейс користувача, який був створений з урахуванням сучасних принципів UI/UX дизайну. Головною особливістю є його адаптивність, що забезпечує коректне відображення та зручність використання на різних пристроях та розмірах екрана. Додаток підтримує два основні візуальні режими: світлий (light) для комфортної роботи протягом дня та темний (dark) для зниження навантаження на очі в умовах низького освітлення. Перемикання між цими темами здійснюється за допомогою спеціального елемента, розташованого у нижньому кутку екрана, що забезпечує гнучкість у персоналізації користувацького досвіду.

Інтерфейс також передбачає зручне меню облікових записів, яке стає доступним після натискання на відображену адресу гаманця. Це меню дозволяє користувачу легко перемикатися між різними обліковими записами MetaMask, а також відключати гаманець від застосунку. Такий підхід значно покращує навігацію та управління обліковими записами, мінімізуючи потребу у взаємодії безпосередньо з розширенням MetaMask для базових операцій. Візуальні приклади головного інтерфейсу застосунку у світлій та темній темах представлені на рисунках 3.3 та 3.4 відповідно. Це демонструє, як зміна теми впливає на загальний вигляд, зберігаючи при цьому чіткість та читабельність інформації.

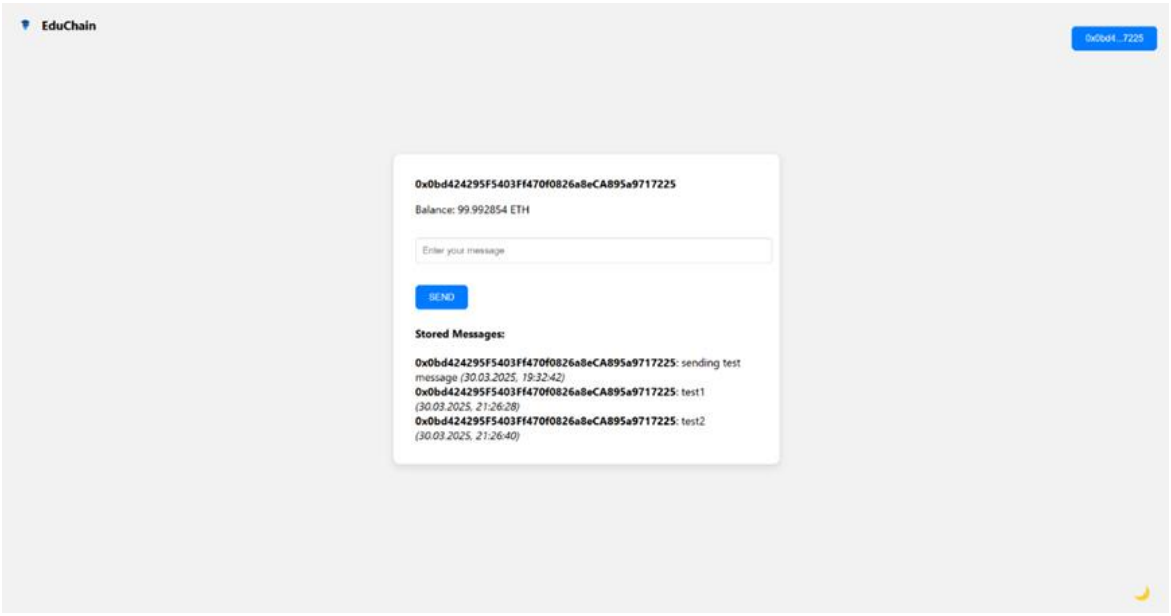

Рисунок 3.3 – Головний інтерфейс додатку у світлій темі

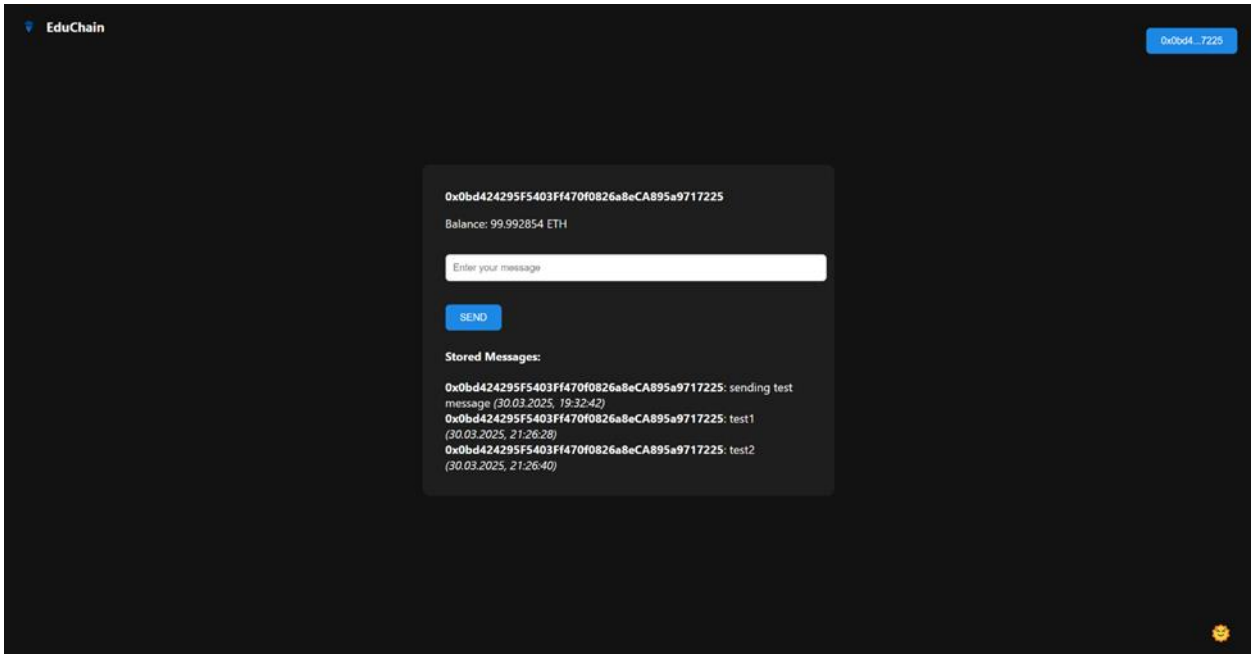

Рисунок 3.4 – Головний інтерфейс додатку у темній темі

4 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Мета тестування

Метою тестування є всебічна та комплексна перевірка функціональності всіх програмних компонентів розробленої системи токенізації освітніх активів. Це включає глибокий аналіз коректності роботи смарт-контракту, який є основою зберігання та управління освітніми записами в блокчейні. Окрім того, тестування спрямоване на виявлення можливих помилок у логіці клієнтської частини, що відповідає за взаємодію користувача з системою.

Ключовим завданням є забезпечення бездоганної та коректної взаємодії між клієнтським інтерфейсом, бібліотекою Web3.js та блокчейн-мережею Ethereum. Це передбачає перевірку правильності надсилання та отримання даних, коректності обробки транзакцій, а також стабільності функціонування застосунку в різних сценаріях використання. Всі ці кроки мають на меті гарантувати високу надійність, безпеку та зручність використання розробленого програмного забезпечення для токенізації освітніх активів.

4.2 Тестування смарт-контракту

Тестування смарт-контракту Storage.sol було проведено в локальному середовищі розробки, використовуючи інструменти Ganache та Truffle Console. Цей підхід дозволив симулювати блокчейн-мережу та взаємодіяти з контрактом у контрольованих умовах, забезпечуючи швидку ітерацію та налагодження. Основні сценарії перевірки були спрямовані на підтвердження коректності функціонування методів контракту та його стійкості до некоректних вхідних даних.

Тестування включало такі ключові сценарії, що охоплювали як успішні

операції, так і обробку виняткових ситуацій. Перевірялося, чи коректно записується повідомлення до блокчейну та чи викликається при цьому відповідна подія MessageStored при надсиланні коректного повідомлення. Очікуваний результат – повідомлення успішно записано, а подія викликана. Також тест перевіряв обробку спроби відправити порожній рядок, де очікуваним результатом було виникнення помилки EmptyMessage(). Сценарій надсилання надто довгого повідомлення передбачав відправлення повідомлення, довжина якого перевищує максимально дозволену межу MAX_MESSAGE_LENGTH. Очікуваним результатом було виникнення помилки MessageTooLong(). Перевірялося, чи функція getMessages() повертає повний та коректний масив із усіма збереженими даними, де очікуваним результатом було повернення масиву зі збереженими даними. Також тест перевіряв, чи функція getMessageCount() повертає точне число збережених повідомлень, де очікуваним результатом було коректне число. Всі ці тести були успішно пройдені.

Результати створення та тестування смарт-контракту в Truffle Console, що ілюструють ці сценарії, наведені на рисунках 4.1 та 4.2. Вони демонструють, що всі визначені тестові випадки були успішно пройдені, підтверджуючи надійність та передбачуваність роботи смарт-контракту Storage.sol. Це є критично важливим для забезпечення цілісності та безпеки освітніх активів у блокчейні.

```
2_deploy_contracts.js
=====================

   Deploying 'Storage'
   -------------------
   > transaction hash:    0x3926ecd2f017036903e7e39f57a68115c3bfa7948872811a73220a6a122f3ffc
   > Blocks: 0            Seconds: 0
   > contract address:    0x51DB3Bdc3ee9F0ee4679311BcdfEa00646E66f47
   > block number:        1
   > block timestamp:     1750108774
   > account:             0xCd15cC1D7DfC9b6c85Cc5bEa011F241E924976F4
   > balance:             99.9928538
   > gas used:            357310 (0x573be)
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.0071462 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:           0.0071462 ETH

Summary
=======
> Total deployments:   1
> Final cost:          0.0071462 ETH
```

Рисунок 4.1 – Результати створення смарт-контракту у Truffle Console

```
Listening on 127.0.0.1:8545
eth_blockNumber
net_version
eth_accounts
eth_getBlockByNumber
eth_accounts
net_version
eth_getBlockByNumber
eth_getBlockByNumber
net_version
eth_getBlockByNumber
eth_estimateGas
net_version
eth_blockNumber
eth_getBlockByNumber
eth_estimateGas
eth_getBlockByNumber
eth_gasPrice
eth_sendTransaction

  Transaction: 0x3926ecd2f017036903e7e39f57a68115c3bfa7948872811a73220a6a122f3ffc
  Contract created: 0x51db3bdc3ee9f0ee4679311bcdfea00646e66f47
  Gas usage: 357310
  Block Number: 1
  Block Time: Tue Jun 17 2025 00:19:34 GMT+0300 (Eastern European Summer Time)
```

Рисунок 4.2 – Результати тестування смарт-контракту у Truffle Console

4.3 Тестування клієнтської частини

Тестування React-додатку проводилося вручну у веб-браузерах Chrome та Firefox, щоб забезпечити кросбраузерну сумісність та належний користувацький досвід. Для симуляції блокчейн-мережі та взаємодії з нею використовувалась локальна Ethereum-мережа Ganache у поєднанні з розширенням MetaMask. Це дозволило перевірити функціональність застосунку в умовах, максимально наближених до реального використання, але без витрат реальних коштів на комісії. Основні перевірені функції включали підключення гаманця, відображення балансу, надсилання та фільтрацію повідомлень, а також перемикання теми інтерфейсу.

Першим етапом тестування була перевірка процесу підключення гаманця MetaMask. Очікувалося, що після натискання на кнопку "Connect Wallet" застосунок успішно ініціює підключення до розширення MetaMask у браузері. Після авторизації користувача в MetaMask, інтерфейс застосунку повинен був відобразити скорочену адресу підключеного облікового запису. Цей сценарій був успішно пройдений, що підтверджує коректність початкової ініціалізації Web3.js та взаємодії з гаманцем. Результат підключення облікового запису MetaMask представлений на рисунку 4.3.

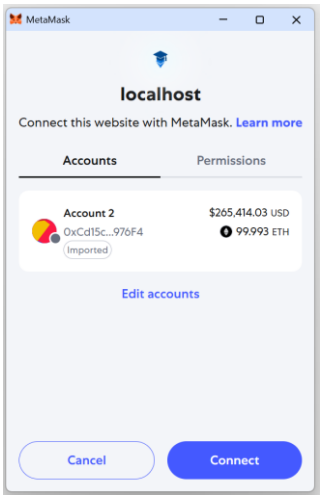

Рисунок 4.3 – Підключення аккаунту MetaMask

Після успішного підключення гаманця наступним кроком було перевірити коректність відображення балансу поточного облікового запису користувача у ETH. Очікувалося, що застосунок точно отримає баланс з блокчейну та відобразить його в інтерфейсі у зручному для читання форматі (ETH). Тестування підтвердило правильне отримання та відображення значення балансу. Показ балансу поточного облікового запису відображено на рисунку 4.4.

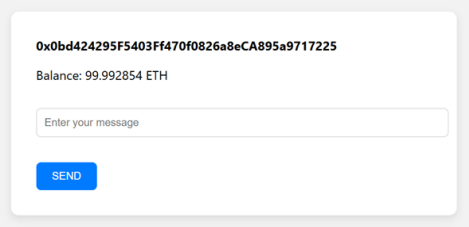

Рисунок 4.4 – Показ балансу поточного акаунта

Ключова функціональність застосунку – надсилання текстового повідомлення до смарт-контракту. Сценарій тестування передбачав введення тексту у відповідне поле та натискання кнопки "SEND". Очікувалося, що після підтвердження транзакції через MetaMask, повідомлення буде успішно збережено в блокчейні та з'явиться у списку відображуваних повідомлень в інтерфейсі користувача, зображено на рисунку 4.5. Цей процес перевіряє повний цикл взаємодії з контрактом: від ініціації транзакції до її підтвердження та оновлення інтерфейсу, зображено на рисунку 4.6.

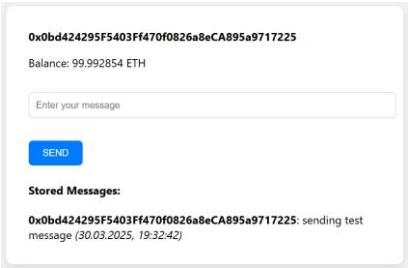

Рисунок 4.5 – Відображуване повідомлення в інтерфейсі користувача

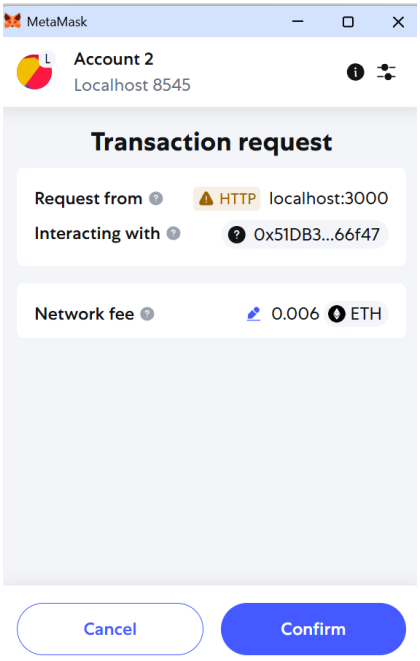

Рисунок 4.6 – підтвердження транзакції через MetaMask

Була перевірена можливість перемикання між різними обліковими записами MetaMask без необхідності відключатися та підключатися заново. Очікувалося, що після вибору іншого акаунта з меню (яке з'являється після кліку на поточну адресу гаманця), інтерфейс автоматично оновить відображувану адресу облікового запису, його баланс та список повідомлень, фільтруючи їх за новим активним акаунтом. Цей сценарій був успішно реалізований, що значно підвищує зручність використання для користувачів з кількома обліковими записами. Список для перемикання акаунтів показано на рисунку 4.7.

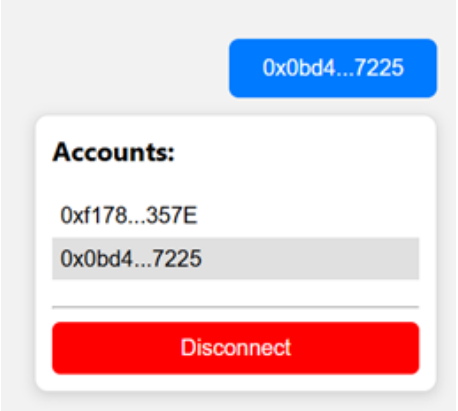

Рисунок 4.7 – Список для перемикання акаунтів

Перевірялася функція відключення поточного облікового запису від застосунку. Після натискання кнопки "Disconnect" (Відключити) очікувалося, що застосунок очистить усі дані, пов'язані з гаманцем (адреса, баланс, повідомлення), та поверне інтерфейс до початкового стану (з кнопкою "Connect Wallet"). Цей тест також був успішно пройдений, демонструючи коректне керування сесією користувача. Процес відключення гаманця можна побачити на рисунку 4.8.

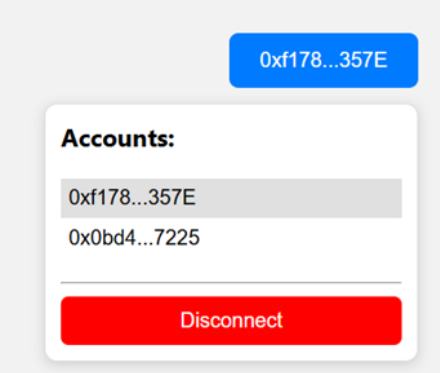

Рисунок 4.8 – Відключення гаманця

Важливим сценарієм була перевірка того, що застосунок коректно фільтрує та виводить лише ті повідомлення, які були надіслані поточним підключеним обліковим записом. Очікувалося, що незалежно від кількості повідомлень у блокчейні, відображатимуться лише релевантні для активного користувача. Ця фільтрація працювала вірно, забезпечуючи конфіденційність та порядок у відображенні даних. Виведення збережених повідомлень окремого користувача проілюстровано на рисунку 4.9.

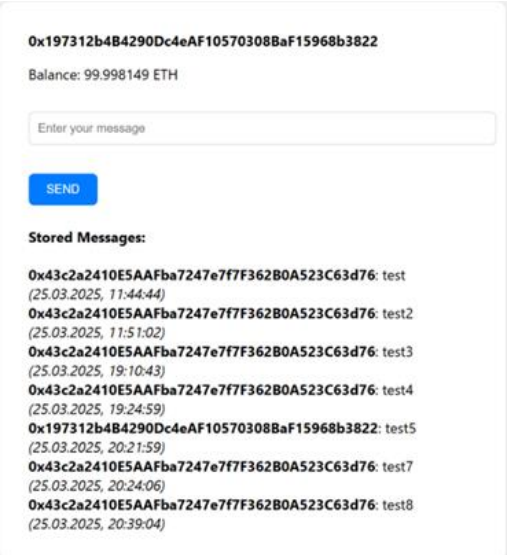

Рисунок 4.9 – Виведення збережених повідомлень окремого користувача

4.4 Результат тестування

Комплексне тестування розробленого децентралізованого застосунку (dApp) підтвердило його стабільну роботу та повну відповідність вимогам технічного завдання. Усі ключові функції, реалізовані як у смарт-контракті, так і в клієнтській частині, були успішно перевірені в середовищі розробника.

Зокрема, тестування продемонструвало коректність взаємодії зі смарт-контрактом, де операції з запису та отримання повідомлень до блокчейну виконуються без помилок. Це включає обробку коректних даних, а також належне реагування на виняткові ситуації, такі як спроба надсилання порожнього або надто довгого повідомлення, що викликає відповідні помилки контракту. Також було підтверджено надійне підключення та управління гаманцем MetaMask: система успішно підключається до MetaMask, відображає актуальний баланс облікового запису користувача та коректно обробляє перемикання між різними обліковими записами, а також їх відключення. Функціональність інтерфейсу користувача була повністю перевірена: адаптивний дизайн та перемикання тем (світла/темна) працюють належним чином, забезпечуючи комфортну взаємодію для користувача, а

відображення повідомлень користувача відбувається з вірною фільтрацією, показуючи тільки релевантні записи. Жодних критичних помилок чи збоїв у логіці взаємодії між клієнтською частиною (React.js), бібліотекою Web3.js та смарт-контрактом виявлено не було.

Загалом, проведене тестування засвідчило високу надійність та зручність використання розробленого програмного забезпечення. Прототип системи продемонстрував свою готовність до подальшого використання та масштабування, підтверджуючи потенціал для токенізації освітніх активів на платформі Ethereum.

4.5 Підсумки четвертого розділу

У рамках четвертого розділу було проведено всебічне тестування розробленого програмного забезпечення, що охопило як смарт-контракт, так і клієнтську частину застосунку. Метою тестування було підтвердження функціональності компонентів, виявлення потенційних помилок у логіці та забезпечення коректної взаємодії із блокчейн-мережею Ethereum.

Тестування смарт-контракту Storage.sol здійснювалося в локальному середовищі з використанням Ganache та Truffle Console. Всі основні сценарії, включаючи надсилання коректних повідомлень, а також обробку порожніх та надто довгих повідомлень, були успішно пройдені. Це підтвердило надійність та передбачуваність роботи контрактної логіки.

Клієнтська частина, реалізована на React.js, тестувалася вручну у браузерах Chrome та Firefox із залученням локальної мережі Ganache та розширення MetaMask. Була перевірена коректність підключення та відображення балансу гаманця, успішне надсилання повідомлень, функціональність перемикання та відключення облікових записів, а також коректна робота перемикача тем інтерфейсу. Особлива увага приділялася фільтрації повідомлень, що виводяться лише для поточного користувача.

Результати тестування свідчать про стабільну роботу як смарт-

контракту, так і клієнтської частини застосунку. Виявлені обмеження, наприклад, щодо довжини повідомлення, коректно обробляються, що дозволяє стверджувати про надійність та зручність використання розробленого програмного забезпечення. Жодних критичних помилок у логіці взаємодії між компонентами виявлено не було. Прототип продемонстрував свою готовність до подальшого використання та масштабування, що підтверджує його потенціал для токенізації освітніх активів.

ВИСНОВКИ

У межах виконання кваліфікаційної роботи на тему "Дослідження, проєктування та розробка програмних компонентів для токенізації освітніх активів на основі смарт-контрактів Ethereum" було досягнуто поставлених цілей і виконано всі основні завдання дослідження.

У теоретичній частині проведено аналіз сучасного стану розвитку блокчейн-технологій, зокрема Ethereum, а також їх можливостей для впровадження у сфері освіти. Досліджено поняття токенізації, смарт-контрактів, принципів роботи децентралізованих застосунків (dApp), криптографічного захисту даних, функціонування алгоритмів хешування, консенсусу, а також основи використання Web3-технологій.

У процесі дослідження було проєктовано та реалізовано архітектуру децентралізованої системи для токенізації освітніх активів. До складу системи входить смарт-контракт на мові Solidity, який забезпечує зберігання структурованих повідомлень у блокчейні, та клієнтська частина на основі React.js, яка відповідає за взаємодію користувача з блокчейном через Web3.js і MetaMask. Користувачі можуть надсилати повідомлення, переглядати історію повідомлень, перемикати акаунти, а також змінювати тему інтерфейсу.

Реалізовано підтримку сучасного дизайну з використанням адаптивного інтерфейсу, стилізацією через styled-components, анімацією через framer-motion та збереженням налаштувань у localStorage. Забезпечено повну інтеграцію з гаманцем MetaMask, що дозволяє здійснювати підпис транзакцій і взаємодію зі смарт-контрактом без посередників.

У ході тестування було перевірено коректність виконання основних функцій застосунку, включаючи підключення гаманця, запис та отримання повідомлень, обробку виключень, перемикання акаунтів, зміну теми тощо. Результати тестування засвідчили стабільну роботу системи та її готовність

до подальшого використання і масштабування.

Практична цінність розробленої системи полягає у можливості створення основи для впровадження децентралізованих освітніх платформ, де освітні активи можуть бути представлені у вигляді токенів або записів у блокчейні. Запропонований підхід дозволяє забезпечити підвищений рівень прозорості, захищеності та верифікованості освітньої інформації.

У перспективі роботу можна розширити шляхом інтеграції стандартів ERC-721 або ERC-1155 для повноцінної токенізації сертифікатів, дипломів або досягнень, зберігання метаданих у IPFS, реалізації авторизації викладачів, додавання підтримки ролей і прав доступу, а також включення функцій аналітики та візуалізації даних.

Таким чином, кваліфікаційна робота повністю відповідає поставленим завданням, є актуальною з наукової та практичної точки зору та демонструє приклад ефективного використання технологій блокчейн у сфері освіти.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ethereum Foundation. Ethereum Whitepaper [Електронний ресурс]. – Режим доступу: https://ethereum.org/en/whitepaper/

2. Buterin V. A Next-Generation Smart Contract and Decentralized Application Platform [Електронний ресурс]. – Ethereum Foundation, 2014.

3. Wood G. Ethereum: A Secure Decentralised Generalised Transaction Ledger – Yellow Paper, 2021. – Режим доступу: https://ethereum.github.io/yellowpaper/paper.pdf

4. ConsenSys. Web3.js Documentation [Електронний ресурс]. – Режим доступу: https://web3js.readthedocs.io/

5. Truffle Suite Documentation [Електронний ресурс]. – Режим доступу: https://trufflesuite.com/docs/

6. Solidity Documentation [Електронний ресурс]. – Режим доступу: https://docs.soliditylang.org/

7. MetaMask Documentation [Електронний ресурс]. – Режим доступу: https://docs.metamask.io/

8. OpenZeppelin Contracts Library [Електронний ресурс]. – Режим доступу: https://docs.openzeppelin.com/contracts

9. Christoph Jentzsch. Decentralized Autonomous Organization to Automate Governance. – 2016.

10. Szmigiera M. Blockchain – Statistics & Facts [Електронний ресурс]. – Statista. – 2023. – https://www.statista.com/topics/5122/blockchain/

11. Lin I. C., Liao T. C. A Survey of Blockchain Security Issues and Challenges // IJ Network Security. – 2017. – Vol. 19, No. 5. – P. 653–659.

12. Zhang Y., Xue R. Security and Privacy on Blockchain // ACM Computing Surveys. – 2019. – Vol. 52, No. 3. – P. 1–34.

13. Tapscott D., Tapscott A. Blockchain Revolution: How the Technology Behind Bitcoin and Other Cryptocurrencies is Changing the World. – Penguin,

2016. – 368 с.

14. Yaga D., Mell P., Roby N., Scarfone K. Blockchain Technology Overview. – National Institute of Standards and Technology, U.S. Department of Commerce. – 2018.

15. Шило С. Blockchain у вищій освіті: аналіз можливостей і викликів // Освіта і суспільство. – 2022. – № 4. – С. 45–52.

16. Курило В. С. Децентралізація даних в освітній сфері // Вісник педагогіки і психології. – 2021. – № 5. – С. 38–43.

17. Мережа Ethereum. Як працює та де використовується [Електронний ресурс]. – https://forklog.com/

18. Ethereum.org – Education Portal [Електронний ресурс]. – https://ethereum.org/en/developers/docs/