МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітня програма: «Безпека комп'ютерних систем і мереж»

Група: 4КБ-01

Дипломний проект

здобувача освіти денної форми навчання
КБ.01.11.000.ДП

КУСТУРОВ АНДРІЙ
СТАНІСЛАВОВИЧ

м. Одеса
2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітня програма: «**Безпека комп'ютерних систем і мереж»**

Група: **4КБ-01**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи) на тему:

Розробка скрипта взаємодії з блокчейн-мережами

Проектний матеріал складається з пояснювальної записки на 88 сторінках та графічного (презентаційного) матеріалу на 13 аркушах (слайдах).

Дипломник _____ (Кустуров А.С.)

Керівник _____ (Стайкуца С.В.)

Консультанти:

з економічної частини _____ (Іванченков В.С.)

з охорони праці _____ (Чорновол Н.І.)

з дотримання вимог ЄСКД _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Скорнякова О.В.)

Завідувач відділення _____ (Кривченко Ю.В.)

Захист «17» 06 2024 р. Протокол ЕК № 1

Оцінка ЕК 4 (чотири) 80 з.

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Безпека комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР *Беркань І.В.*

"16" 01 2024 р.

ЗАВДАННЯ

на дипломний проект

Здобувачеві освіти *Кустурова Андрія Станіславовича*

(прізвище, ім'я, по батькові)

1. Тема проекту *Розробка скрипта взаємодії з блокчейн-мережами*

затверджена наказом по коледжу від "02" листопада 2023 р. № 244-А2-ОД

2. Термін здачі закінченого проекту 13.06.24р

3. Вихідні данні до проекту *1. Дослідити основну інформацію щодо технології блокчейн; 2. Дослідити інформацію щодо криптовалютних гаманців та смарт контрактів; 3. Розглянути технології консенсусу; 4. Типи мереж - L1 та L2; 5. Розробити скрипт взаємодії з блокчейн-мережами*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Аналіз щодо технології блокчейн; Аналіз технології шифрування; Аналіз технологій консенсусу; Дослідження видів та алгоритмів блокчейнів; Дослідження класифікації стандартів токенів; Види криптовалютних гаманців; Незручність у процесі взаємодії з блокчейном; розробка скрипту для автоматизації взаємодії з блокчейном

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Приклад з'єднання мереж; Приклад потенційного використання блокчейна; Технології консенсусу; Мережі типу L1(Layer 1); Проблеми мереж типу L1(Layer 1); Мережі типу L2(Layer 2); Порівняння трьох платформ блокчейна за кількістю транзакцій та вартістю цих транзакцій; Класифікація стандартів токенів; Характеристики та методи користування смартконтрактами; Криптовалютні гаманці; Види децентралізованих додатків; Схема скрипту та його результат роботи.

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Стайкуца С.В*		
Економічний розділ	*Іванченков В.С.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання _____

Керівник *Стайкуца С.В.* _____
(підпис)

Завдання прийняв до виконання *Кустуров А.С.* _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	*Вступ. Постановка задачі проектування*	*29.04.24-2.05.24*	виконав
2	*Аналіз технічного завдання та пошук літератури*	*2.05.24-4.05.24*	виконав
3	*Аналіз технологій блокчейну*	*4.05.24-10.05.24*	виконав
4	*Аналіз технологій шифрування*	*10.05.24-15.05.24*	виконав
5	*Дослідження видів та алгоритмів блокчейнів*	*15.05.24-19.05.24*	виконав
6	*Дослідження класифікації стандартів токенів*	*19.05.24-23.05.24*	виконав
7	*Розробка скрипта для автоматизації взаємодією з*	*19.05.22-25.05.24*	виконав
	блокчейн мережами		
8	*Випробування застосунку та аналіз результатів*	*25.05.24-29.05.24*	виконав
9	*Виконання економічних розрахунків*	*29.05.24-2.06.24*	виконав
10	*Розробка питань з охорони праці та техніки безпеки*	*2.06.24-6.06.24*	виконав
11	*Підготовка мультимедійної презентації проекту*	*6.06.24-9.06.24*	виконав

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КБ 01. 11 000. 00 ДП	Дипломний проект		
А4			КБ 01. 11 000. 00 ДП ПЗ	Пояснювальна записка	1	
			КБ 01. 11 000. 00 ДП МП	Мультимедійна презентація		

Зм.	Арк.	№ докум.	Підпис	Дата	КБ 01. 11 000. 00 ДП		
Розробив		Кустуров А.С.			Розробка скрипта взаємодії з блокчейн-мережами	Літ.	Аркуш 4 / Аркушів
Перевірив		Стайкуца С.В.				Н Д П	
Н. Контр.		Петрашова В.І.				ВСП "ОТФК ОНТУ" гр.4КБ-01	
Затверд.		Кривченко Ю.В.					

ЗМІСТ

ВСТУП

Технологія блокчейн стала однією з найвпливовіших інновацій у сфері цифрових технологій за останнє десятиліття. Своїм походженням блокчейн завдячує криптовалютам, зокрема Bitcoin, але його потенціал широко виходить за межі фінансового сектору. Завдяки децентралізації, надійності та прозорості, блокчейн знаходить застосування в логістиці, охороні здоров'я, голосуванні та інших ключових областях. Однак, із зростанням використання виникають питання масштабованості та ефективності, які розглядаються через використання різноманітних алгоритмів консенсусу: Proof of Work, Proof of Stake, Delegated Proof of Stake, та інші. Кожен з них вносить свої корективи в безпеку, швидкість обробки та енергоефективність системи. Ці та інші технічні нововведення, такі як криптографічне хешування та асиметричне шифрування, сприяють створенню безпечного та ефективного цифрового середовища. Від початкового проектування та вибору відповідної архітектури до кінцевого впровадження і експлуатації, кожен етап розробки блокчейн-системи вимагає глибокого аналізу та адаптації під конкретні потреби використання.

1 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

1.1 Фундаментальні відомості щодо технології блокчейн

1.1.1 Визначення та суть блокчейна

Блокчейн являє собою структуровану послідовність блоків, які формуються за строго визначеними правилами і зберігають інформацію. Копії цих ланцюгів блоків розміщуються на численних комп'ютерах, які оперують незалежно один від одного. Це забезпечує відмінний рівень резервування даних, адже у разі виходу з ладу кількох систем, загальний обсяг даних залишається неторканим.

В структурі кожного блока присутній часовий відбиток, хеш попереднього блока та інформація про транзакції, що сприяє захисту від фальсифікацій та несанкціонованого доступу. Блокчейн конструйований таким чином, що кожен новий блок містить посилання на свого попередника, тому будь-які спроби змінити інформацію в одному з блоків вимагають змін у всіх наступних блоках, значно ускладнюючи несанкціоновані втручання.

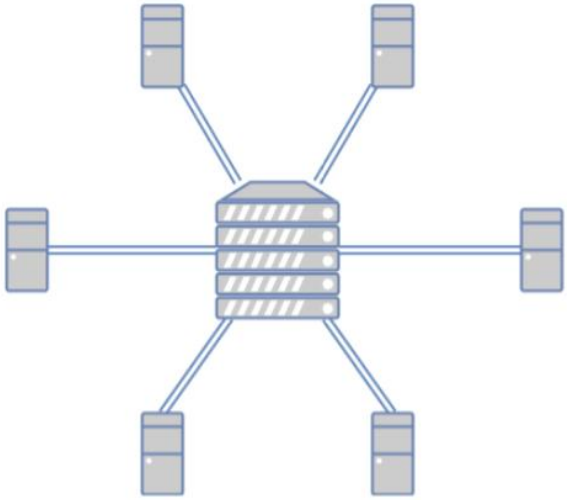


Рисунок 1.1. Традиційна централізована мережа

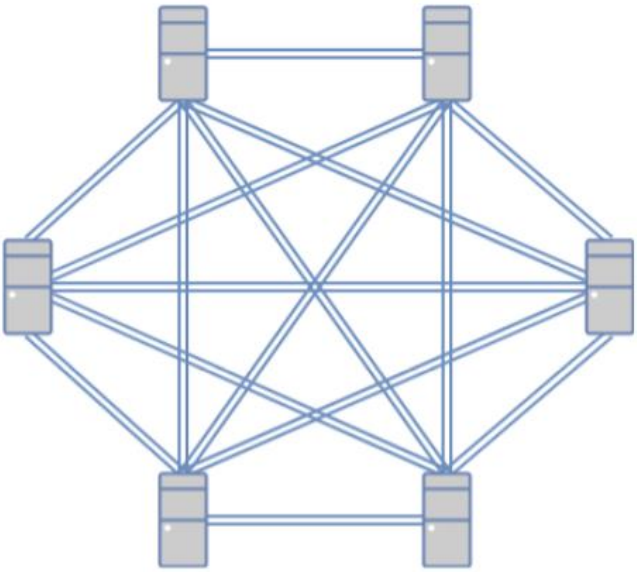


Рисунок 1.2. Розподілена мережа через технологію блокчейн

1.1.2 Основні характеристики блокчейну

Децентралізація: Однією з основних особливостей блокчейна є відсутність централізованого контрольного органу. Це означає, що дані розосереджені серед численних незалежних вузлів, що спільно здійснюють управління та підтримку мережі, забезпечуючи її стійкість до збоїв та зловмисних атак.

Відкритість і труднощі у зміні інформації: Блокчейн відрізняється прозорістю та відкритістю. Кожен блок включає хеш-ідентифікатор попереднього блока, що надає послідовний та незмінний ланцюг. Така структура ускладнює будь-які спроби маніпуляції даними, оскільки для зміни інформації в одному блоку необхідно внести зміни в усі наступні блоки.

Надійність та методи захисту: Захист даних у блокчейні гарантується застосуванням криптографічних технологій, таких як хешування та цифрові підписи. Ці методи допомагають запобігати несанкціонованому доступу та забезпечують високий рівень безпеки збережених даних.

Імунітет до змін: Блокчейн забезпечує цілісність інформації завдяки тому, що записи, однажди створені, не можуть бути змінені або видалені без одностайної згоди учасників мережі. Ця особливість робить блокчейн особливо привабливим для систем, де важлива вірність та незмінність інформації.

1.1.3. Застосування блокчейна у різних областях

Блокчейн має широкий спектр потенційних застосувань, включаючи:

- Фінанси та банківська справа: Децентралізовані криптовалюти можуть змінити спосіб, яким люди зберігають і передають гроші, оминаючи посередників, таких як банки;
- Постачання та логістика: Блокчейн може використовуватись для відстеження поставок, контролю якості та управління ланцюгами поставок з урахуванням достовірності інформації;
- Охорона здоров'я: У медичній сфері блокчейн може забезпечити безпечне зберігання та передачу медичних записів, забезпечуючи конфіденційність та доступність даних;
- Управління правами та голосування: Технологія блокчейн може забезпечити прозорість та достовірність виборів, а також управління правами доступу до цифрових активів та контенту;
- Розумні контракти та децентралізовані програми: Блокчейн дозволяє створювати програмні коди (розумні контракти), які виконуються автоматично при виконанні певних умов, що може бути корисно в різних сферах, від фінансів до автоматизації бізнес-процесів.

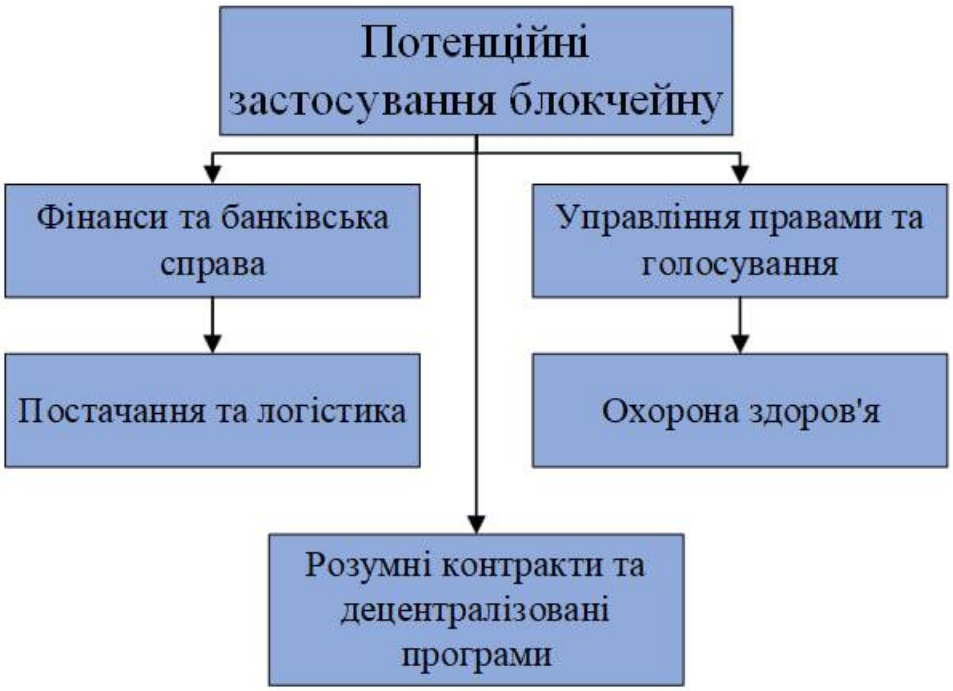


Рисунок 1.3. потенційні застосування блокчейну.

1.1.4 Переваги та виклики блокчейну

Блокчейн, попри свої незаперечні переваги, має ряд викликів і обмежень, які можуть вплинути на його застосування та розвиток. Основними з них є масштабованість, енергоспоживання, пов'язане з деякими алгоритмами консенсусу, а також питання правового регулювання.

Масштабованість: Блокчейн стикається з проблемою масштабованості, особливо в популярних мережах, таких як Bitcoin та Ethereum. Як технологія, заснована на децентралізації, кожен вузол в мережі повинен обробляти велику кількість транзакцій та зберігати повну копію блокчейну, що може призвести до затримок та зниження продуктивності зі зростанням мережі. Вирішення цієї проблеми потребує нових технологічних рішень та підходів, таких як розробка Layer 2 протоколів або шардингу.

Енергоспоживання: Механізм консенсусу Proof of Work (PoW), який використовується в багатьох блокчейнах, вимагає великої кількості обчислювальних ресурсів, що призводить до значного енергоспоживання. Це стало предметом суспільної та екологічної критики, оскільки вуглецевий слід, пов'язаний із майнінгом криптовалют, може бути дуже великим. Відповідно, існує великий інтерес до більш енергоефективних альтернатив, таких як Proof of Stake (PoS).

Правове регулювання: Блокчейн також зіштовхується з викликами в контексті правового регулювання. Через свою новизну і глобальний характер, юридичні норми часто відстають від технологічного розвитку, створюючи правову невизначеність для користувачів, розробників та бізнесів. Це включає питання забезпечення приватності, обігу цифрових активів, та інші регулятивні аспекти, що можуть вплинути на прийняття та використання блокчейну в ширшому масштабі.

1.1.5 Аналіз технологій шифрування

Хешування відноситься до методу обробки даних, де інформація будь-якої довжини перетворюється у фіксований рядок символів стандартної довжини.

Цей процес є фундаментальним для забезпечення цілісності даних у криптографічних застосуваннях та блокчейн-технологіях. Основні атрибути хеш-функцій включають:

Константність розміру результату: Незалежно від розміру вхідного потоку даних, вихід хеш-функції має стандартний, фіксований розмір. Цей аспект є критичним для забезпечення сталості та передбачуваності обробки даних, а також гарантує, що вихід хешу відповідає специфікаціям систем, в яких він використовується.

Суттєвість мінімальних змін (Uniformity): Властивість хеш-функцій полягає у тому, що дрібні зміни в даних ведуть до помітних і суттєвих змін у результаті хешування. Ця характеристика значно ускладнює підбір чи відгадування вхідних даних на основі знання хешу, тим самим підвищуючи рівень безпеки.

Оперативність обчислень: Хеш-функції оптимізовані для швидкої обробки, що дозволяє їм швидко генерувати хеші, навіть при обробці великих обсягів даних. Це особливо важливо для застосувань у реальному часі та системах, де вимагається негайна обробка великої кількості транзакцій, наприклад у блокчейн-технологіях.

Односпрямованість (One-way): Хеш-функції виробляють результати, які не можуть бути перетворені назад на оригінальні дані. Відновлення вихідних даних із хешу практично неможливе без використання масивних обчислювальних ресурсів або спеціалізованих методик, таких як брутфорс.

Прикладом алгоритму хешування, що широко використовується, є SHA-256 (Secure Hash Algorithm 256-bit). У контексті блокчейна хешування застосовується для створення унікального ідентифікатора (хешу) для кожного блоку даних. Кожен блок містить хеш попереднього блоку, що забезпечує цілісність даних: навіть невелика зміна вмісту блоку призведе до значної зміни його хеша, що робить маніпуляції даними вкрай складними.

Асиметричне шифрування, також відоме як криптографія з відкритим ключем, покладається на два ключі: відкритий ключ і закритий ключ. Відкритий

ключ використовується для шифрування даних, а закритий ключ використовується для їх розшифровки. Хоча ці ключі математично пов’язані, фактично неможливо вивести приватний ключ із відкритого. Асиметричне шифрування має такі особливості:

- Безпека: Закритий ключ потрібен для розшифровки даних, забезпечуючи безпеку передачі конфіденційної інформації;
- Підпис даних: відкритий ключ також можна використовувати для перевірки підпису, створеного закритим ключем. Це дозволяє всім учасникам мережі перевіряти автентифікацію даних;
- Цифрові підписи: Алгоритми асиметричного шифрування також використовуються для створення цифрових підписів, які можна використовувати для автентифікації та перевірки авторства вмісту.

Приклади асиметричного сполучення включають алгоритм RSA та алгоритм цифрового підпису еліптичної кривої або ECDSA. Асиметричне шифрування використовується для захисту транзакцій і передачі даних між учасниками мережі блокчейн.

Симетричне шифрування, з іншого боку, використовує один секретний ключ для кодування та декодування даних. Це означає, що і відправнику, і одержувачу потрібен доступ до одного секретного ключа. Симетричне шифрування має такі особливості:

- Продуктивність: Симетричні алгоритми зазвичай швидші та менш ресурсомісткі;
- Ефективність: симетричне шифрування добре підходить для масового шифрування даних;
- Керування ключами: ключі повинні безпечно передаватись між сторонами, що може бути складно під час обміну даними в мережі.

Приклади симетричних пар включають Advanced Encryption Standard або AES і Data Encryption Standard або DES. У блокчейні симетричне шифрування можна використовувати для захисту даних, як-от ідентифікаційна інформація користувача, що зберігається на платформі. Ці криптографічні методи

відіграють вирішальну роль у захисті блокчейну та забезпеченні його безпеки, цілісності та надійності через безпеку даних та обмін між мережами.

1.1.6. Технології консенсусу

Принцип роботи Proof of Work (PoW) у тому, що учасники мережі, звані майнерами, вирішують математично складні завдання, щоб додати нові блоки в ланцюжок транзакцій. Ці завдання, як правило, потребують обчислювальних ресурсів та енергії для їх вирішення. Коли майнер вирішує завдання, він створює новий блок, який потім додається до блокчейну.

Переваги Proof of Work (PoW):

- Безпека: PoW забезпечує високий рівень безпеки, оскільки для атаки на мережу необхідно контролювати більшість обчислювальних потужностей мережі;
- Децентралізація: Оскільки майнінг доступний для будь-якого учасника мережі, PoW сприяє децентралізації блокчейну.
- Прозорість: Усі учасники мережі можуть легко перевірити правильність вирішення задачі PoW;

Недоліки Proof of Work (PoW):

- Енерговитратність: Майнінг в PoW вимагає значної кількості енергії, що може бути невигідно з екологічної точки зору;
- Масштабованість: Підвищення складності PoW може призвести до проблем масштабованості та збільшення часу обробки транзакцій;
- Централізація майнінгу: У PoW спостерігається централізація майнінгу в руках великих майнінгових ферм із високими обчислювальними потужностями.

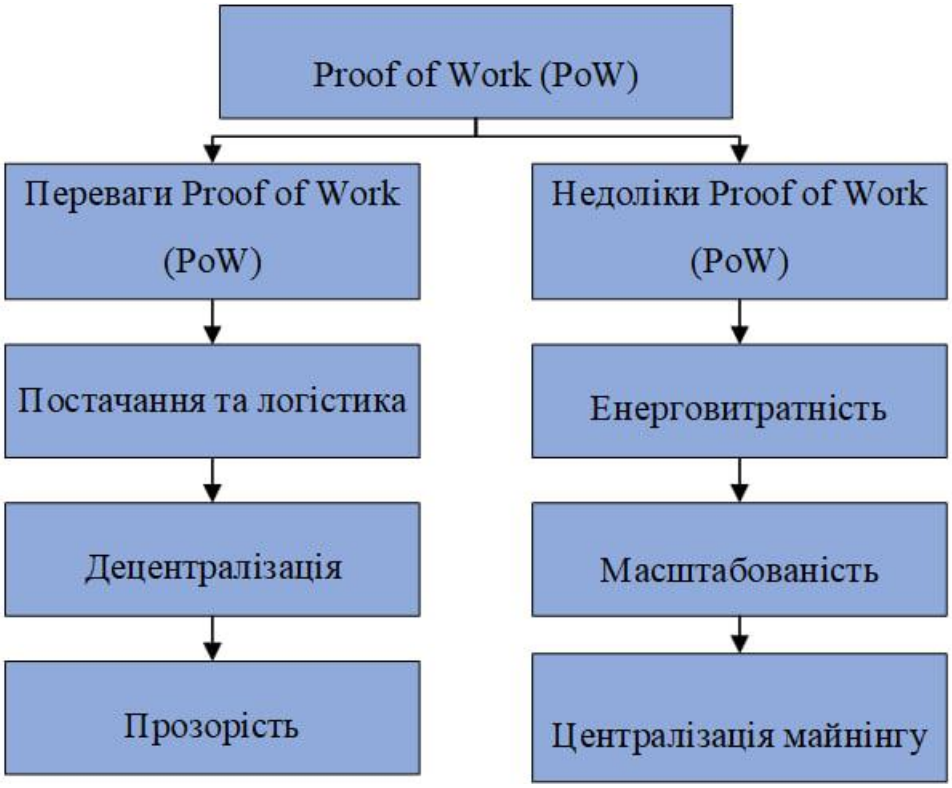


Рисунок 1.4. Переваги та недоліки Proof of Work (PoW).

Новий блок створюється майнером, який першим розв'яже задачу PoW. Після того, як майнер вирішить завдання, він збирає транзакції з пулу, що чекають на підтвердження, створює новий блок і транслює його в мережу. Інші учасники мережі перевіряють правильність рішення та валідність транзакцій, і якщо блок проходить перевірку, він додається до ланцюжка.

Завдання PoW полягає у знаходженні такого числа (nonce), при додаванні якого до блоку та обчисленні його хеша, вийде число з певною кількістю провідних нулів. Це вимагає безліч спроб, оскільки хеш-функція видає псевдовипадкові результати, і майнери повинні пробувати різні значення nonce до того часу, доки знайдуть правильне рішення.

Складність алгоритму PoW регулюється протоколом блокчейна підтримки стабільного часу створення блоків. Якщо блоки створюються дуже швидко, складність збільшується, щоб ускладнити завдання PoW. Якщо блоки створюються надто повільно, складність зменшується. Це регулювання дозволяє мережі підтримувати стабільну продуктивність.

Приклади мереж, які використовують алгоритм Proof of Work:

- Bitcoin (BTC): Перша та найвідоміша криптовалюта, заснована на алгоритмі PoW;
- Litecoin (LTC): Криптовалюта, заснована на алгоритмі PoW, який схожий на Bitcoin, але має більш короткий час створення блоків.

В цілому, Proof of Work (PoW) є ефективним та надійним методом консенсусу, який забезпечує безпеку та децентралізацію у блокчейні. Однак його недоліки, такі як енерговитратність та централізація майнінгу, стимулюють пошук альтернативних рішень.

В алгоритмі Proof of Stake (PoS) нові блоки створюються не шляхом виконання обчислювально складних завдань, як у PoW, а шляхом "ставки" (стейкінгу) певної кількості монет у блокчейні. Чим більше монет учасник ставить у заставу (ставить на стейкінг), тим більша ймовірність, що його блок буде обраний для додавання в блокчейн. Це робить майнінг більш економічно ефективним та екологічно стійкішим, ніж PoW.

Переваги Proof of Stake (PoS):

- Енергоефективність: PoS вимагає значно менше енергії, ніж PoW, оскільки немає потреби у вирішенні обчислювальних завдань;
- Децентралізація: PoS також сприяє децентралізації мережі, оскільки майнінг доступний для будь-якого учасника мережі, який має достатньо монет для стейкінгу;
- Масштабованість: PoS зазвичай більш масштабуємо, ніж PoW, оскільки немає потреби у великих обчислювальних ресурсах.

Недоліки Proof of Stake (PoS):

- Централізація багатства: PoS може сприяти накопиченню монет у невеликої кількості учасників мережі, що може призвести до централізації багатства;
- "Нічим не забезпечені" атаки: Учасник, який контролює більшість монет у мережі, може спробувати атакувати систему.

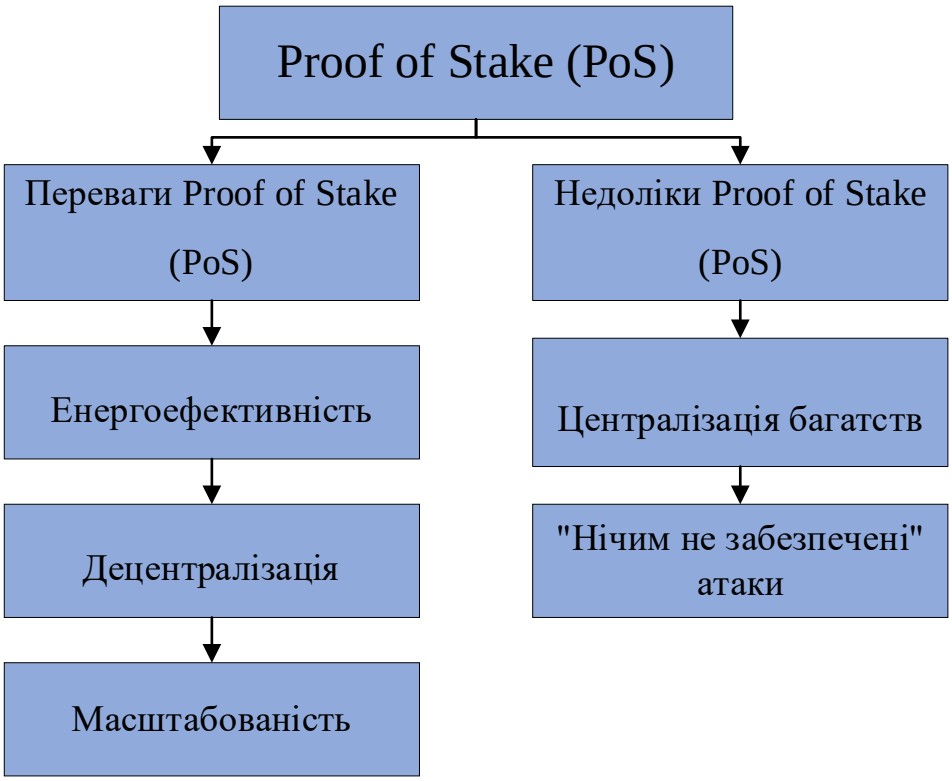


Рисунок 1.5. Переваги та недоліки Proof of Stake (PoS).

Новий блок у PoS створюється майнером, який вибирається випадково з урахуванням кількості монет, які він поставив на стейкінг. Чим більше монет у майнера у стейкінгу, тим більше його шансів бути обраним.

Завдання в PoS полягає в тому, щоб вибрати випадкового учасника мережі для створення нового блоку на основі його частки у загальній кількості монет у стейкінгу.

Складність PoS переважно залежить від кількості монет, поставлених на стейкінг. Однак може існувати додаткова складність, наприклад, облік віку монет (кількість часу, що вони перебували на стейкінгу), що може впливати на можливість вибору майнера для створення нового блоку.

Приклади мереж, які використовують алгоритм Proof of Stake:

- Ethereum 2.0 (ETH): Ethereum PoS з реалізацією Ethereum 2.0, що покращило ефективність та екологічну стійкість мережі;
- Cardano (ADA): Cardano використовує PoS у своїй блокчейн-платформі з метою забезпечення безпеки та децентралізації;
- Tezos (XTZ): Tezos – ще одна блокчейн-платформа, яка використовує PoS для консенсусу.

Proof of Stake є енергоефективним та екологічно стійким методом консенсусу, який забезпечує безпеку та децентралізацію в блокчейні. Однак існують деякі виклики, такі як можливість централізації багатства та "нічим не забезпечені" атаки, які мають бути враховані та вирішені для успішної реалізації PoS.

В алгоритмі Delegated Proof of Stake (DPoS) учасники мережі обирають делегатів (зазвичай званих свідками чи довіреними особами), які мають право створювати нові блоки та проводити транзакції. Замість того, щоб кожен учасник міг ставити свої монети на стейкінг, учасники делегують свої монети делегатам, довіряючи їм створення блоків та проведення транзакцій.

Переваги Delegated Proof of Stake (DPoS):

- Ефективність: DPoS ефективніший, ніж PoW та PoS, оскільки учасники делегують свої монети тим, кого вважають найбільш надійними, замість того щоб виконувати обчислювальні завдання або тримати великі суми монет на стейкінгу;
- Масштабованість: DPoS зазвичай більш масштабуємо, ніж PoW та PoS, завдяки участі обмеженої кількості делегатів у створенні блоків;
- Децентралізація: Хоча DPoS може бути менш децентралізованим порівняно з PoW та PoS через обмежену кількість делегатів, він все одно сприяє децентралізації, оскільки делегати обираються голосами учасників.

Недоліки Delegated Proof of Stake (DPoS)

- Централізація контролю: У DPoS влада зосереджена в руках щодо небагатьох делегатів, що може призвести до централізації контролю над мережею;
- Можливість впливу зацікавлених сторін: Великі володарі монети можуть маніпулювати вибором делегатів, щоб захистити свої інтереси, що може спотворювати принципи децентралізації.

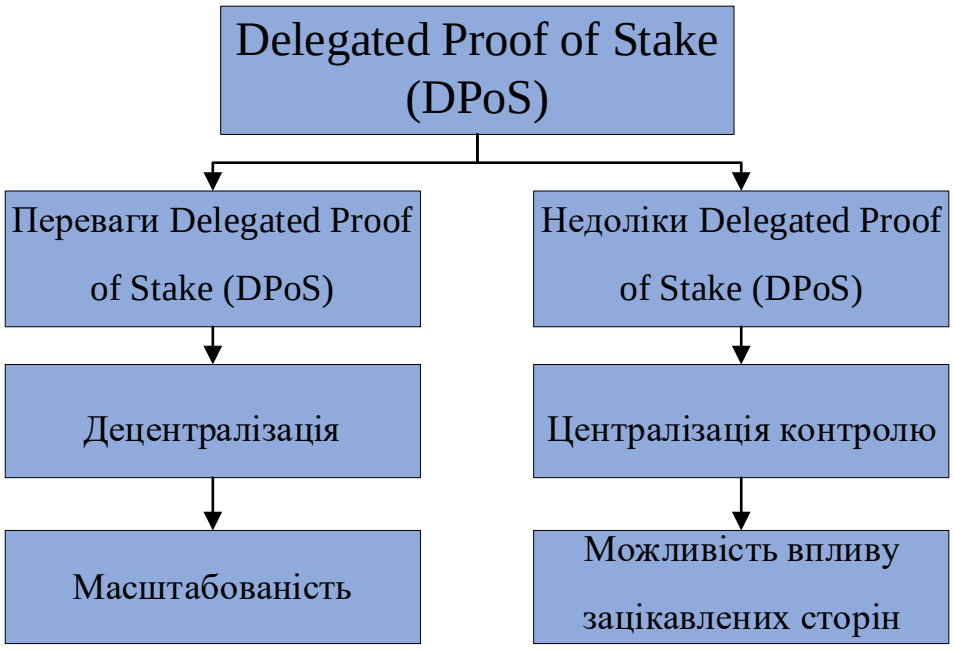


Рисунок 1.6. Переваги та недоліки Delegated Proof of Stake (DPoS)

Новий блок у DPoS створюється делегатами, обраними учасниками мережі на основі їх кількості делегованих монет. Делегати вносять нові транзакції до блоку та підписують його, після чого блок додається до ланцюжка.

Завдання DPoS полягає в тому, щоб забезпечити створення нових блоків та проведення транзакцій у мережі за допомогою вибраних делегатів, довірених учасниками мережі.

Складність DPoS негаразд явно визначається, як у PoW чи PoS. Вона залежить від процесу вибору делегатів та їхнього відповідального управління мережею.

Приклади мереж, які використовують алгоритм Delegated Proof of Stake:

- EOS: EOS, блокчейн-платформа, використовує DPoS для консенсусу, де 21 делегат мають право створювати блоки;
- Tron (TRX): Tron також використовує DPoS, де 27 делегатів створюють нові блоки та проводять транзакції.

Delegated Proof of Stake (DPoS) є ефективним і масштабованим методом консенсусу, який використовує голоси учасників мережі для вибору делегатів, відповідальних за створення блоків і проведення транзакцій. Однак, хоча DPoS може бути більш ефективним і швидким, ніж інші алгоритми консенсусу, він

також схильний до ризику централізації контролю та маніпуляцій з боку великих власників монет.

1.1.7 Дослідження видів та алгоритмів блокчейнів

Мережі першого рівня (Layer 1, L1) є базовими блокчейнами, які забезпечують необхідну інфраструктуру для проведення транзакцій з криптовалютами. Ці мережі функціонують як децентралізовані реєстри, що фіксують усі транзакції на незмінному блокчейні. Таким чином, мережі 1 рівня гарантують безпеку, прозорість і децентралізацію криптовалютних систем. Крім того, вони підтримують основні операції різних додатків і протоколів, надаючи надійне та масштабоване середовище для розвитку блокчейн-індустрії та криптовалютного сектора.

Архітектура мережі:

L1 мережі зазвичай мають однорангову архітектуру, де кожен вузол мережі (комп'ютер) зберігає копію блокчейна. Вузли мережі взаємодіють один з одним для перевірки транзакцій, додавання нових блоків блокчейн і забезпечення консенсусу.

Механізм консенсусу:

А так само, як було зазначено в розділі 1.3, L1 мережі використовують різні механізми консенсусу для досягнення згоди між вузлами мережі про те, які транзакції дійсні і повинні бути додані в блокчейн.

Найбільш поширені механізми консенсусу:

- Proof of Work (PoW);
- Proof of Stake (PoS);
- Delegated Proof of Stake (DPoS).

Менш поширені механізми консенсусу:

- Proof of Authority (PoA) — це консенсусний алгоритм, який базується на репутації валідаторів у мережі. Ці валідатори, які мають бути попередньо перевірені, відповідають за створення блоків та забезпечення безпеки блокчейну. PoA розроблений для ефективності

та економії енергії, пропонуючи альтернативу таким методам як Proof of Work (PoW) та Proof of Stake (PoS) ;

– Proof of Staked Authority (PoSA) — це консенсусний алгоритм, який комбінує елементи Proof of Authority (PoA) та Proof of Stake (PoS). Він був спеціально розроблений для Binance Smart Chain і має на меті забезпечити швидкі та ефективні транзакції з низькими витратами на енергію.

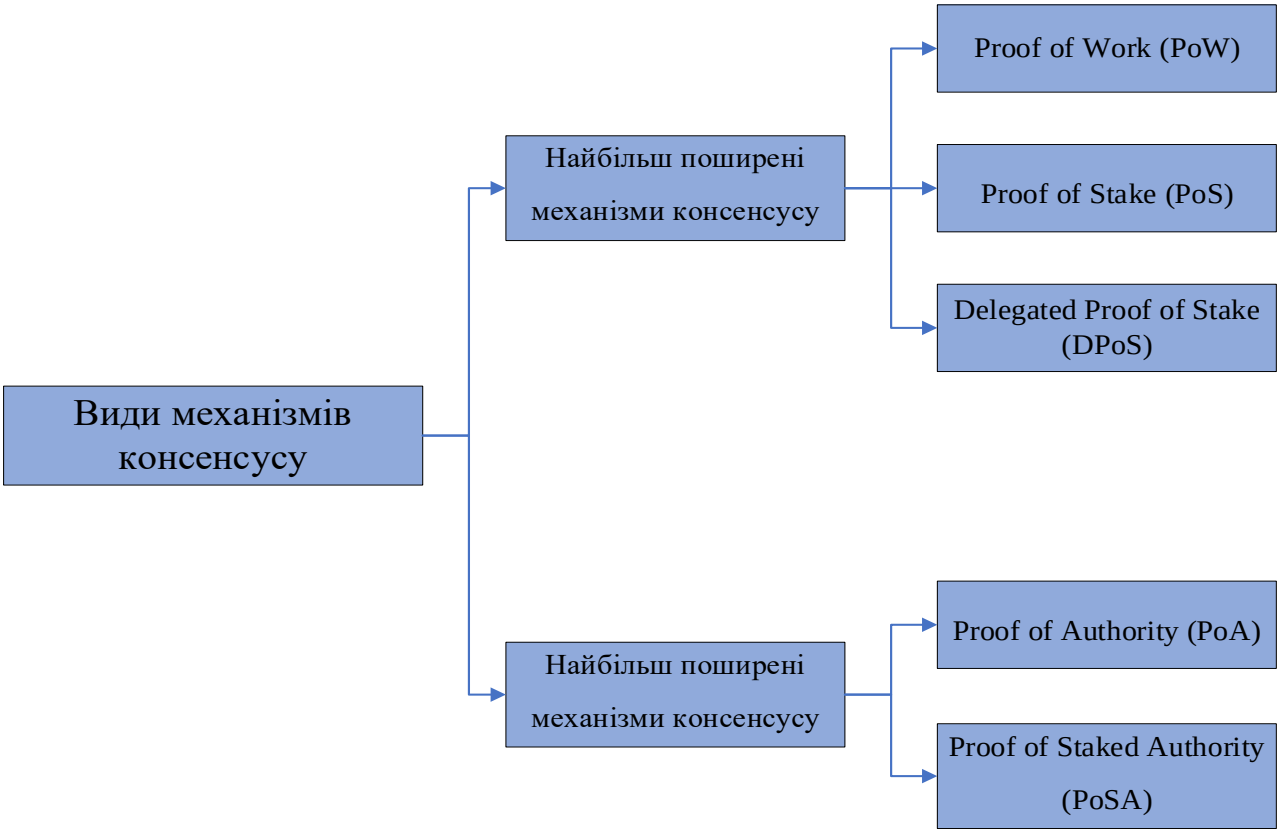


Рисунок 1.7. Механізми консенсусу у блокчейні.

Переваги:

– Безпека: L1 мережі забезпечують високий рівень безпеки завдяки криптографічним методам та децентралізованій архітектурі;

– Прозорість: Всі транзакції в мережах L1, що забезпечує прозорість і підзвітність;

– Децентралізація: L1 мережі не контролюються якоюсь однією стороною, що робить їх більш стійкими до цензури та маніпуляцій;

– Універсальність: L1 мережі можуть використовуватися для різних цілей, таких як переказ коштів, створення смарт-контрактів, децентралізовані програми (dApps) та багато іншого.

Недоліки:

– Масштабованість: мережі L1 можуть мати обмежену пропускну здатність, що може призвести до затримок в обробці транзакцій;
– Енергоспоживання: Деякі механізми консенсусу, такі як PoW, можуть споживати значну кількість енергії.

Приклади:

– Bitcoin: Перший та найвідоміший L1 блокчейн, що використовує PoW консенсус;
– Ethereum: L1 блокчейн, який підтримує смарт-контракти, використовував PoW (до The Merge) та PoS (після The Merge) ;
– Solana: L1 блокчейн, що використовує консенсус PoS;
– Binance Smart Chain (BSC): Блокчейн, що використовує консенсус PoSA для оптимізації швидкості та вартості транзакцій.

Порівняльний аналіз:

В таблиці 1.1 представлено порівняння основних параметрів блокчейн-платформ таких як Bitcoin, Ethereum, Solana та Binance Smart Chain. Аналіз включає тип консенсусу, приблизну максимальну пропускну здатність (TPS), у деяких випадках приблизний час створення блоку та комісію за транзакцію для кожної з платформ.

Таблиця 1.1. Порівняння характеристик різних блокчейн-платформ

Назва	Тип консенсусу	Приблизна максимальна пропускна здатність (TPS)	Час створення блоку	Комісія за транзакцію
Bitcoin	PoW	7 транзакцій/сек	10 хвилин	$20 - $90
Ethereum	PoS	20-45 транзакцій/сек	12 секунд	$2 - $110
Solana	PoS	5000-7000 транзакцій/сек	0.4 секунди	$0.01 - $0.05
Binance Smart				

Chain (BSC)	PoSA	200 транзакцій/сек	3 секунди	$0.01 - $1

Мережі другого рівня (L2) розроблені як доповнення до первинних блокчейнів (L1), таких як Bitcoin або Ethereum, і спрямовані на покращення їхніх характеристик за рахунок збільшення пропускної спроможності, зниження вартості транзакцій та підвищення рівня конфіденційності. Важливо зазначити, що мережі L2 не функціонують незалежно, а взаємодіють з базовими блокчейнами L1, забезпечуючи цим безпеку та децентралізацію.

Мережі L2 ефективно відсікають частину транзакцій з головного блокчейну, пересилаючи їх на додатковий рівень. Цей процес значно знижує навантаження на базову мережу, що сприяє збільшенню її пропускної здатності та зменшенню витрат на транзакції. Таке рішення може знизити вартість проведення транзакцій до менш ніж одного цента, замість декількох доларів, які були б типовими для первинного блокчейну.

L2 (Layer 2) рішення не використовують власних технологій консенсуса так, як це робить основний блокчейн (L1). Замість цього, вони залежать від механізму консенсуса основної мережі для забезпечення безпеки та децентралізації.

Архітектура мережі:

L2 мережі можуть мати різні архітектури, але зазвичай вони використовують один із таких підходів:

- Пакунки (Rollups): Об'єднують безліч транзакцій в одну, що знижує навантаження на L1;
- Канали стану (State channels): Дозволяють сторонам здійснювати транзакції між собою без необхідності запису кожної транзакції в блокчейн;
- Бічні ланцюги (Sidechains): Це окремі блокчейни, пов'язані з L1.

Технологія масштабування:

Optimistic rollups — це метод масштабування блокчейн-мереж, який підвищує їх пропускну спроможність шляхом обробки транзакцій за межами основного ланцюга. Ця технологія базується на "оптимістичному" припущенні, що всі транзакції є правильними за замовчуванням, вимагаючи втручання лише у випадку виявлення помилки або махінацій. Верифікація здійснюється через систему викликів, де транзакції можуть бути оскаржені протягом певного періоду. Це дозволяє значно знизити витрати на газ та покращити загальну пропускну здатність мережі, забезпечуючи при цьому високий рівень безпеки і децентралізації.

ZK rollups є передовою технологією масштабування блокчейн-мереж, яка використовує нульове розкриття інформації (zero-knowledge proofs), спеціально zk-SNARKs, для забезпечення ефективності та конфіденційності транзакцій. Цей метод дозволяє агрегувати велику кількість транзакцій в один згорнутий доказ, який підтверджується і публікується на основному блокчейні. Оскільки перевірка відбувається без необхідності відтворення всіх транзакцій, ZK rollups значно знижують використання ресурсів і зберігають високий рівень безпеки та конфіденційності, одночасно забезпечуючи значно швидшу обробку та меншу вартість транзакцій порівняно з основним ланцюгом.

Переваги:

- Підвищена пропускна здатність: мережі L2 можуть обробляти більше транзакцій в секунду, ніж L1 блокчейни;
- Знижені комісії: мережі L2 можуть пропонувати нижчі комісії, ніж L1 блокчейни;
- Покращена функціональність: мережі L2 можуть додавати нові функції, які недоступні в L1 блокчейнах.

Недоліки:

- Складність використання (на програмному рівні): L2 мережі можуть бути складнішими у використанні, ніж L1 блокчейни;
- Обмежена сумісність: мережі L2 можуть бути сумісні не з усіма L1 блокчейнами;

– Залежність від L1: Безпека мереж L2 залежить від безпеки L1, на якому вони побудовані.

Приклади:

1. Arbitrum: Optimistic rollup рішення для Ethereum;
2. Optimism: Optimistic rollup рішення для Ethereum;
3. zkSync: ZK rollup рішення для Ethereum;
4. StarkNet: ZK rollup рішення для Ethereum.

Перспективи розвитку:

Блокчейн-мережі першого рівня (L1) стикаються з проблемою, відомою як "трилема блокчейна", за якою мережа не може одночасно бути швидкою, безпечною та економічно вигідною. Чим більше безпеки та децентралізації забезпечує мережа, тим менш ефективною стає її пропускна спроможність, що призводить до високих витрат на транзакції та повільної обробки. Мережі другого рівня (L2), такі як Optimistic та ZK rollups, вирішують цю проблему, переміщаючи частину обробки транзакцій за межі основної мережі. Це дозволяє значно збільшити пропускну спроможність без компромісу в безпеці, оскільки основний блокчейн все ще гарантує валідність і безпеку транзакцій. Таким чином, L2 мережі надають швидкість та економічну ефективність, не жертвуючи при цьому безпекою.

Рисунок 1.8. Трилема блокчейна, ілюстрація взаємозалежності між безпекою, масштабуванням та децентралізацією в блокчейн-технологіях.

L2 мережі - це область, що активно розвивається в сфері блокчейн-технологій. Нові технології та рішення постійно розробляються для того, щоб покращити масштабованість, безпеку, конфіденційність та зручність використання L2 рішень. L2 рішення можуть відігравати важливу роль у майбутньому блокчейн-технологій, дозволяючи L1 блокчейнам масштабуватись та обслуговувати мільйони користувачів.

Порівняльний аналіз:

В таблиці 1.2 представлено порівняльний аналіз характеристик різних L2 рішень, зокрема таких як Arbitrum, Optimism, zkSync, StarkNet та Scroll. Для кожної мережі наведено тип rollup, приблизну максимальну пропускну здатність (TPS), приблизний час створення блоку та діапазон комісій за транзакцію. Ці дані дозволяють оцінити ефективність та економічну привабливість кожного з рішень в контексті масштабування блокчейн-технологій.

Таблиця 1.2. Порівняльний аналіз характеристик мереж L2 з різними типами rollups

Мережа	Тип Rollup	Приблизна пропускна здатність (TPS)	Приблизний час створення блоку	Комісія за транзакцію
Arbitrum	Optimistic rollup	4,000 транзакцій/сек	10 секунд	$0.1-$0.5
Optimism	Optimistic rollup	2,000 транзакцій/сек	10 секунд	$0.1-$0.5
zkSync	ZK rollup	10,000 транзакцій/сек	2 секунди	$0.01-$0.05
StarkNet	ZK rollup	8,000 транзакцій/сек	10 секунд	$0.001-$0.005
Scroll	ZK rollup	10,000 транзакцій/сек	2 секунди	$0.01-$0.05

Порівняння трьох платформ блокчейна: Ethereum, Binance Smart Chain (BSC), та Arbitrum, за кількістю транзакцій та вартістю цих транзакцій, становить інтерес для оцінки ефективності та доступності кожної системи. Нижче наведено аналіз цих параметрів для кожного блокчейну:

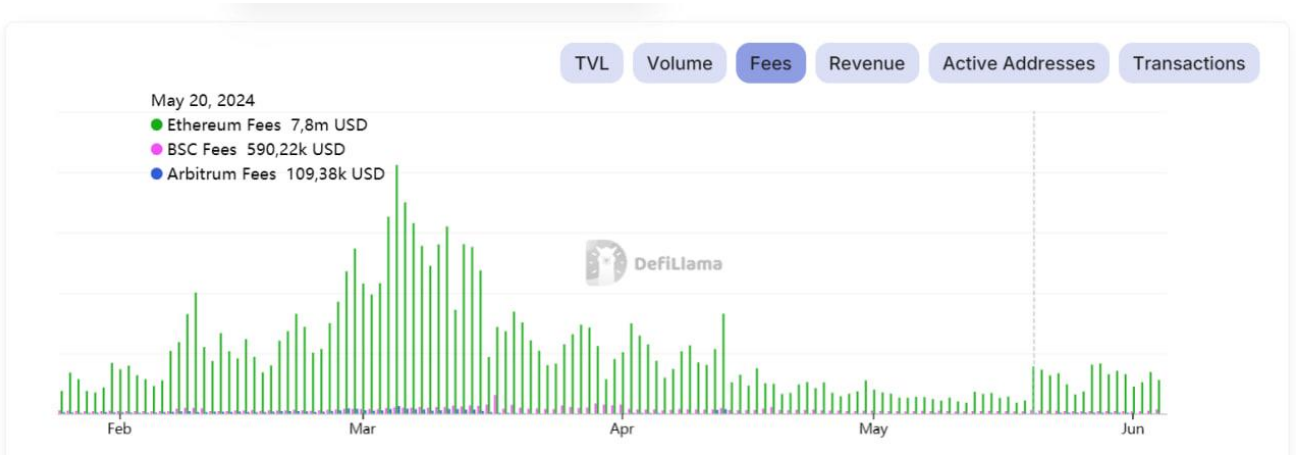


Рисунок 1.8. Порівняння плати за транзакції у трьох різних блокчейнах.

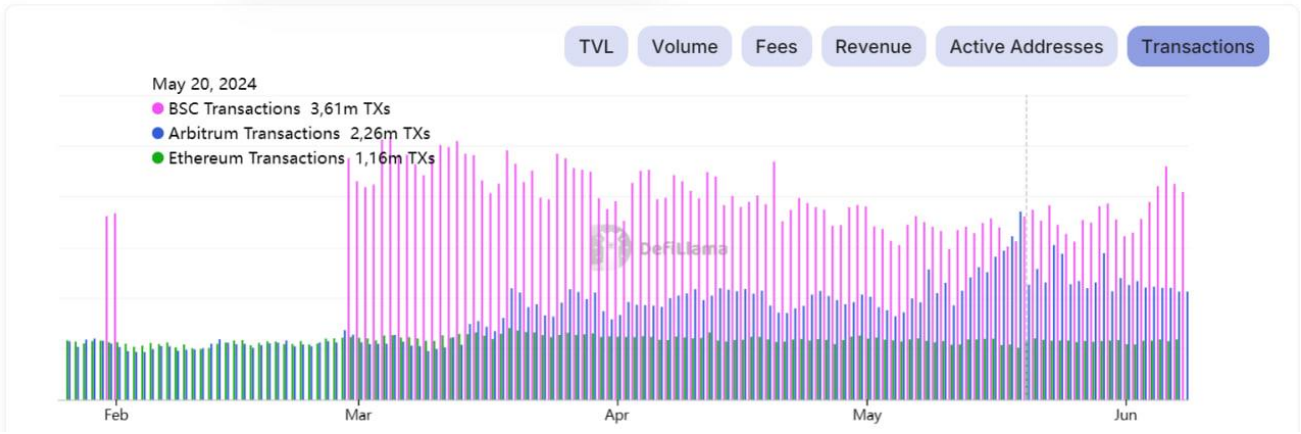


Рисунок 1.9. Кількість транзакцій у трьох різних блокчейнах.

1.1.8 Класифікація стандартів токенів

Токени є фундаментальним елементом у сфері блокчейна та криптовалют, представляючи собою цифрові активи, які можуть бути використані для різноманітних цілей, таких як оплата послуг та участь у голосуванні. Для спрощення створення, використання та взаємодії токенів на блокчейн-платформах існують стандарти токенів. Ці стандарти представляють собою набір правил і технічних вимог, які токени мають дотримуватися для забезпечення сумісності з певною блокчейн-платформою. Вони регулюють

процеси створення, передачі та взаємодії токенів з іншими контрактами та користувачами у мережі.

ERC-20 токен – це стандарт токенів на базі блокчейну Ethereum. Він задає ряд правил і стандартів, яких повинні дотримуватися всі токени, випущені на платформі Ethereum. ERC-20 токени є одними з найпоширеніших типів токенів.

Функціонал:

- Баланс токенів;
- Переклади;
- Дозволи на витрати;
- Події відповідно до рисунку 1.10 .

Приклади:

1. Tether (USDT):
   - Стабільний коін, прив'язаний до вартості долара США.Використовується для торгівлі та збереження вартості в криптовалютних біржах.
2. Chainlink (LINK):
   - Токен платформи децентралізованих оракулів.Використовується для оплати послуг оракулів, що надають дані з зовнішнього світу для смарт-контрактів.

Переваги:

- Простота використання;
- Широка сумісність;
- Підтримка основних гаманців та біржі.

Недоліки:

- Не підходить для незамінних токенів;
- Обмежена функціональність.

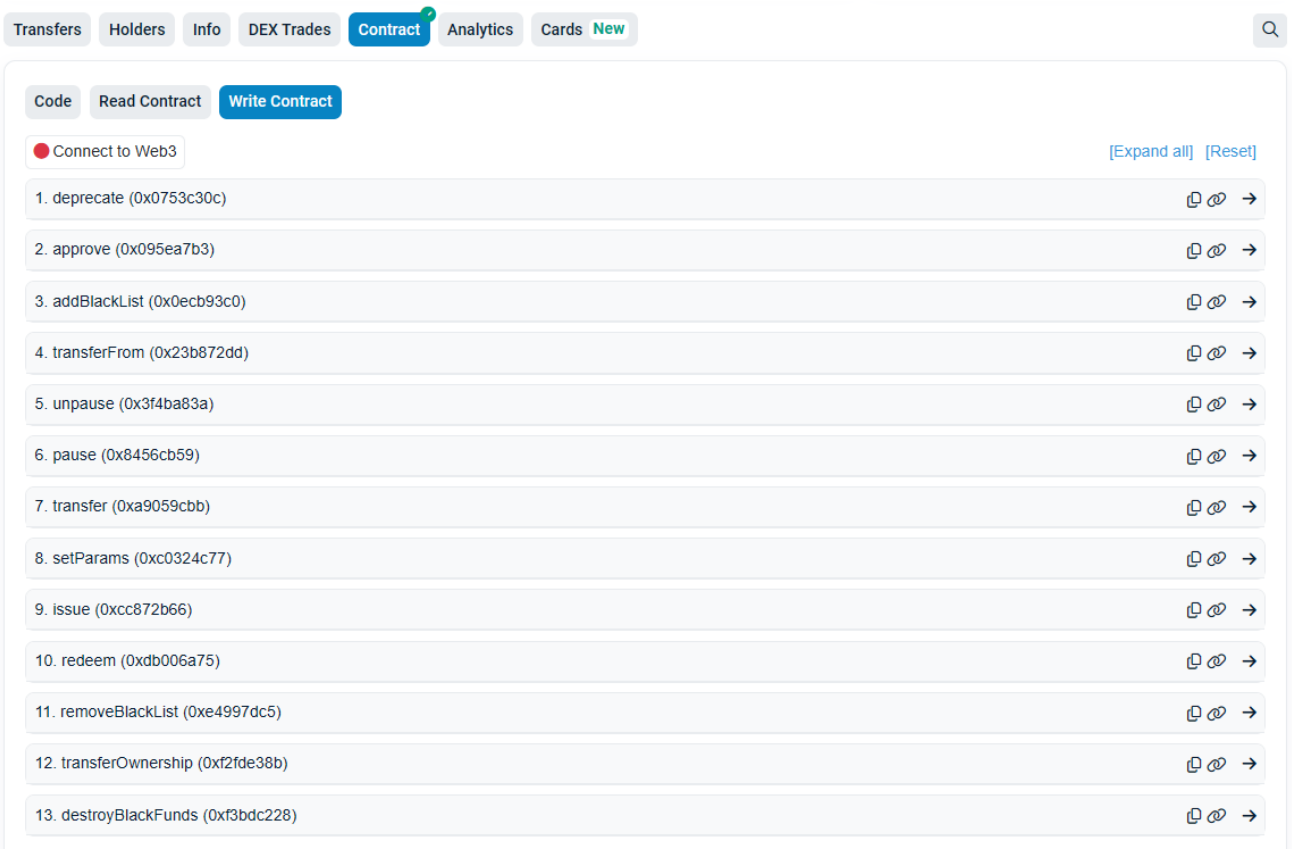


Рисунок 1.10. Функції контракту ERC-20 токена, зосереджені на адміністративних операціях та переказах.

ERC-721 токен – це стандарт невзаємозамінних токенів (NFT) на базі блокчейну Ethereum. На відміну від ERC-20 токенів, які є взаємозамінними (кожен токен еквівалентний іншому), кожен ERC-721 токен є унікальним і має свою індивідуальну вартість і характеристики.

Функціонал:

- Володіння токеном;
- Пересилач маркерів;
- Схвалення на передачу;
- Метадані (наприклад, ім'я, опис);
- Події відповідно до рисунку 1.11.

Приклади:

1. CryptoKitties:

- Колекційні віртуальні котики, кожен з яких є унікальним. Використовуються для розведення, колекціонування та торгівлі на платформі CryptoKitties.

2. Decentraland (LAND):
   - Віртуальні земельні ділянки у віртуальному світі Decentraland. Використовуються для будівництва віртуальних об'єктів і проведення заходів у Decentraland.

Переваги:

- Підходить для унікальних цифрових активів;
- Висока безпека;
- Розширюваність.

Недоліки:

- Більш складний у використанні, ніж ERC-20;
- Менш сумісний з основними гаманцями та біржами.

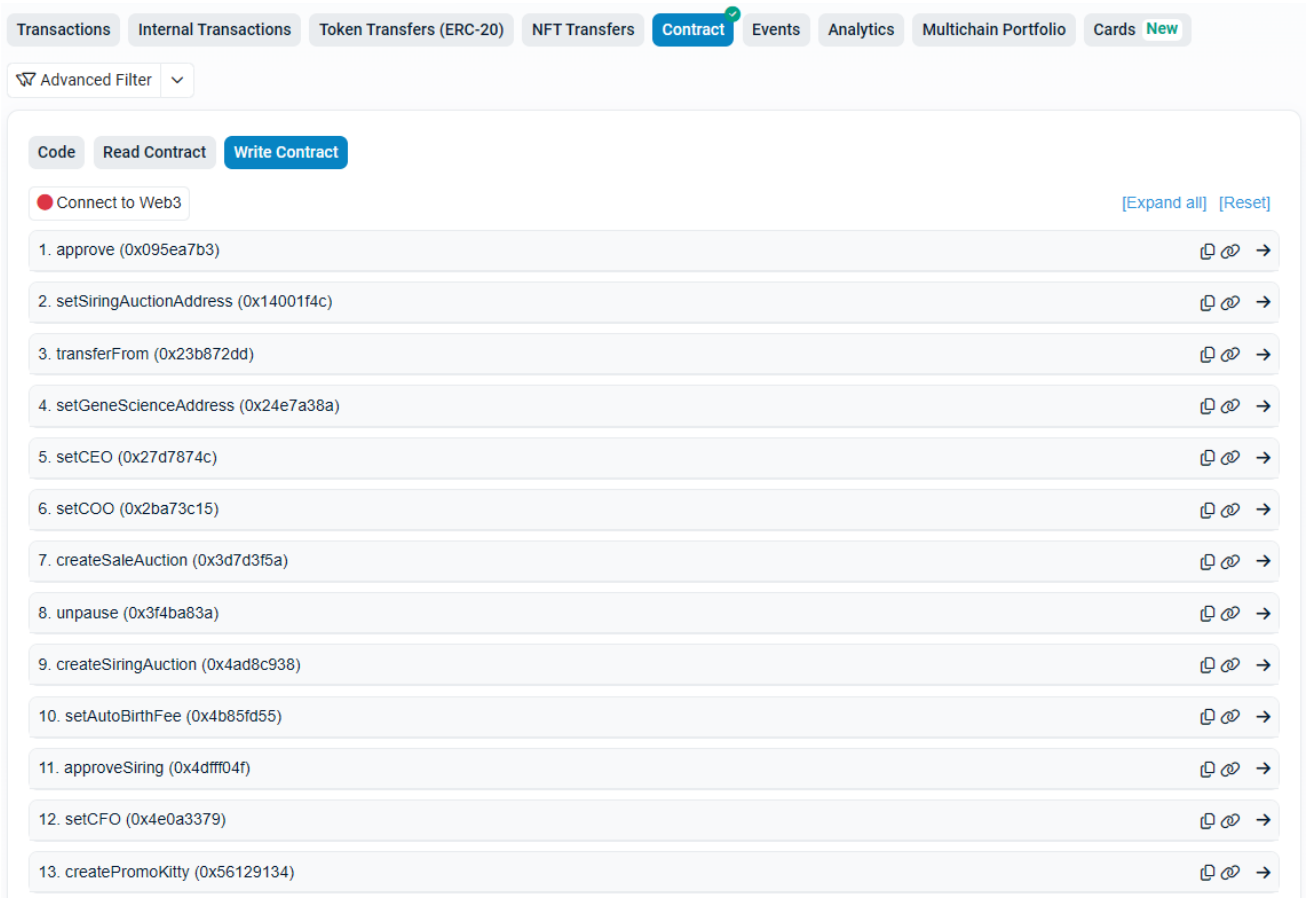


Рисунок 1.11. Функції контракту CryptoKitties ERC-721, зосереджені на управлінні активами та аукціонами.

ERC-1155 є передовим стандартом мульти-токенів для Ethereum блокчейна, який надає можливість одночасного створення фангібельних (взаємозамінних) та незамінних токенів (NFT) у межах одного і того ж смарт-контракту. Цей стандарт значно оптимізує взаємодію між різними типами активів, дозволяючи здійснювати їх передачу та використання більш ефективно та безпечно. Завдяки уніфікації інтерфейсів, ERC-1155 забезпечує значні переваги перед традиційними стандартами, такими як ERC-20, який призначений лише для фангібельних токенів, та ERC-721, який використовується для незамінних токенів. Така інтеграція сприяє зменшенню кількості транзакцій і витрат на газ, оскільки усі операції можуть виконуватись через один контракт. Це не тільки спрощує розробку додатків на блокчейні, але й створює потенціал для нових типів додатків, що можуть використовувати гібридні токени для забезпечення більш широкого спектру функцій і послуг.

Функціонал:

- Баланси замінних та незамінних токенів;
- Переклади;
- Дозволи на витрати;
- Події відповідно до рисунку 1.10.

Приклади:

1. Enjin Coin (ENJ):
   - Платформа для створення і керування ігровими активами. Використовується для створення взаємозамінних та невзаємозамінних токенів в іграх, побудованих на платформі Enjin.
2. Gods Unchained:
   - Карткова гра, в якій картки представлені як ERC-1155 токени. Гравці можуть володіти, торгувати і використовувати ці картки в грі.

Переваги:

- Гнучкість
- Підходить для різноманітних застосувань
- Економія витрат на газ

Недоліки:

- Менш поширений, ніж ERC-20 та ERC-721
- Обмежена підтримка основними гаманцями та біржами.

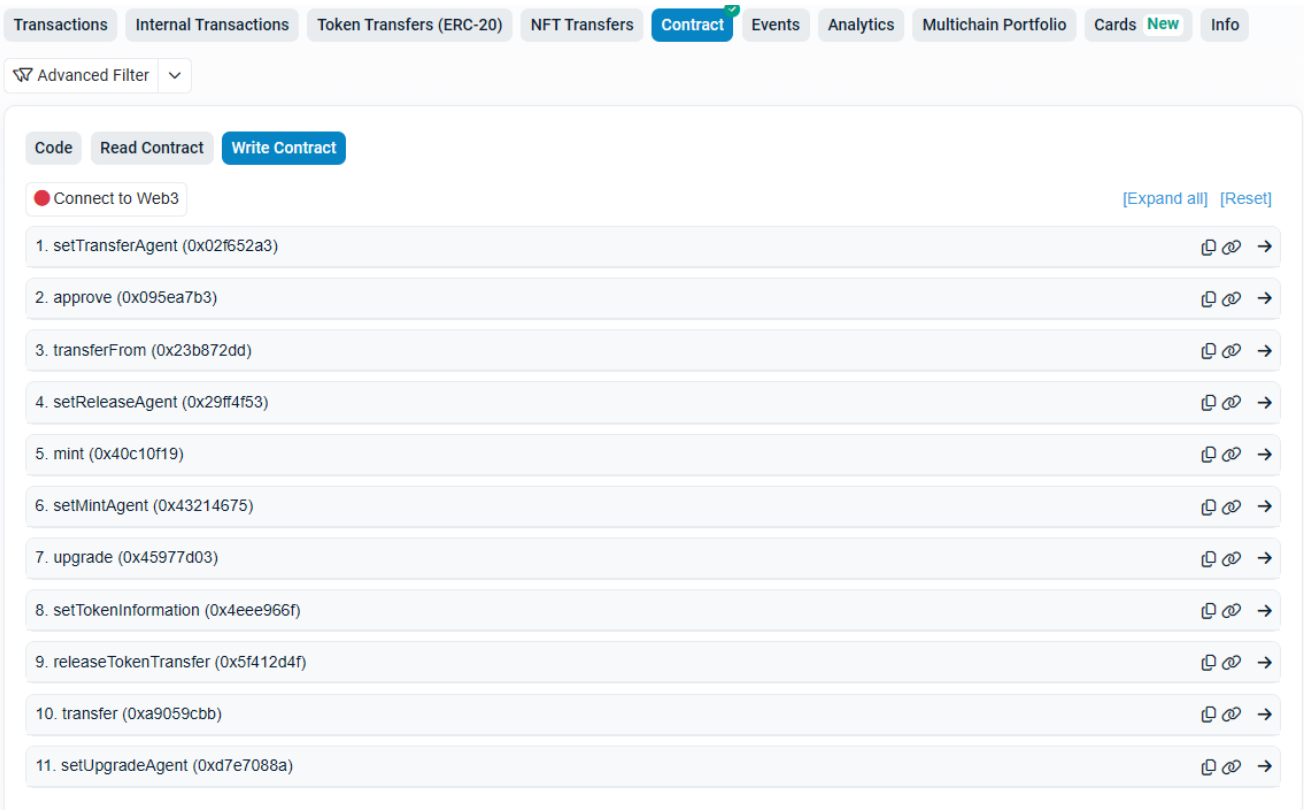


Рисунок 1.12. Функції ERC-1155 токена з можливостями оновлення та мінтингу.

BEP-20 – це стандарт токенів, розроблений для блокчейну Binance Smart Chain (BSC). Він аналогічний стандарту ERC-20 на блокчейні Ethereum, надаючи функціонал для створення та управління взаємозамінними токенами (fungible tokens), які можна ділити на частки.

Функціонал:

- Баланс токенів;
- Переклади;
- Дозволи на витрати;
- Події як зображені на рисунку 1.10.

Приклади:

1. Binance Coin (BNB):
   - Оригінальний токен біржі Binance. Використовується для оплати торгових комісій на Binance, участі в токен-сейлах на Binance Launchpad, а також для стейкінгу і голосування на платформі Binance Smart Chain (BSC).
2. PancakeSwap (CAKE):

– Управлінський токен децентралізованої біржі PancakeSwap. Використовується для стейкінгу, фармінгу, участі в лотереях і голосуванні за зміни в протоколі PancakeSwap.

Переваги:

– Простота використання;
– Широка сумісність;
– Підтримка major wallets and exchanges.

Недоліки:

– Централізація: Binance Smart Chain критикує свою відносну централізацію в порівнянні з Ethereum;
– Залежність від екосистеми BSC: Токени BEP-20 обмежені екосистемою Binance Smart Chain і можуть мати труднощі з інтеграцією в інші блокчейни.

1.1.9 Характеристики та методи користування смартконтрактами

Смарт-контракти – це програмні коди, які автоматично виконують та контролюють умови угоди між двома або більше сторонами за певних умов, заданих заздалегідь. Ці умови можуть бути, наприклад, обмін цифровими активами, виконання послуг або будь-які інші дії, які можуть бути програмно визначені та автоматично виконані.

Основні характеристики смарт-контрактів:

1. Автоматизація: Смарт-контракти виконуються автоматично без необхідності участі третіх сторін або посередників у разі настання певних умов.
2. Надійність: Використання блокчейна забезпечує надійність та несуперечність смарт-контрактів. Якось опубліковані та запущені на блокчейні, вони не можуть бути змінені або видалені.
3. Децентралізація: Смарт-контракти працюють у рамках децентралізованих мереж, таких як Ethereum, що означає відсутність централізованого контролю чи управління.

4. Прозорість: Вся інформація про смарт-контракти та їх виконання доступна для всіх учасників мережі блокчейну та може бути перевірена.
5. Програмування: Смарт-контракти можуть бути запрограмовані для виконання широкого спектру завдань та дій, залежно від потреб та умов угоди.
6. Незалежність від третіх сторін: Оскільки смарт-контракти виконуються автоматично, не потрібно довіряти або покладатися на третіх сторін для виконання угоди.

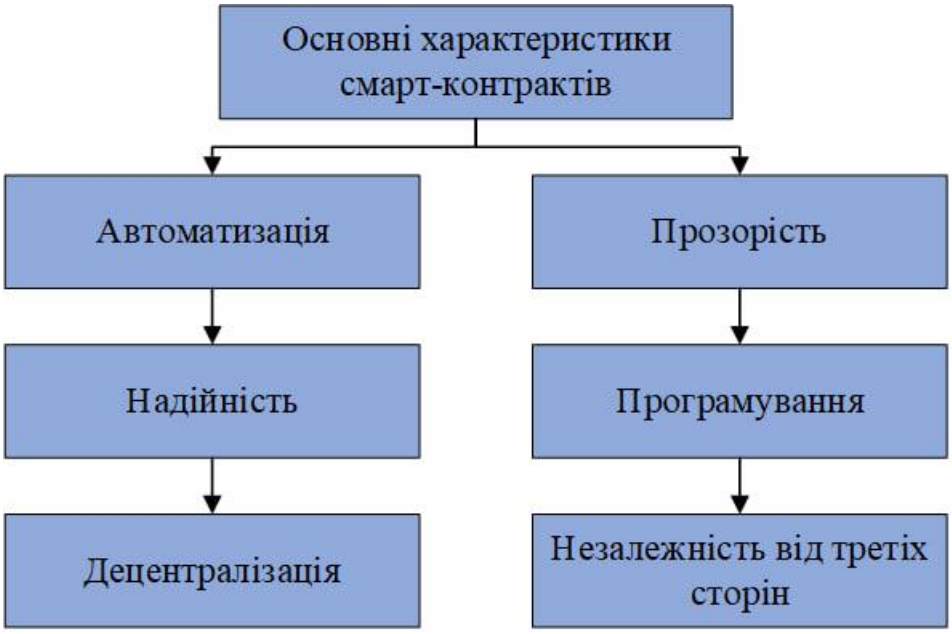


Рисунок 1.13. Основні характеристики смарт-контрактів.

Застосування смарт-контрактів включають області такі як цифрові фінанси (наприклад, криптовалютні транзакції, кредитування, страхування), управління ланцюжками поставок, голосування, нерухомість і багато іншого.

Solidity, створена для виконання на Ethereum Virtual Machine (EVM), є мовою програмування, спеціалізованою на розробці смарт-контрактів, що інтегрує передові характеристики для роботи з децентралізованими додатками (DApps). Однією з важливих особливостей Solidity є її статична типізація, що забезпечує визначення типів усіх змінних на етапі компіляції, що допомагає знизити ризики помилок під час виконання програми.

Solidity підтримує наслідування, що робить можливим створення складних ієрархій контрактів та повторне використання коду, що є істотною перевагою для розробників. Крім того, мова забезпечена багатим набором бібліотек та інтерфейсів, що відкривають широкі можливості для модульного програмування, підвищують читабельність коду та спрощують його підтримку. Система обробки винятків в Solidity дозволяє розробникам безпечно управляти помилками та відхиленнями, забезпечуючи надійність та стабільність виконання контрактів.

Застосування Solidity у сфері блокчейну є різноманітним. Воно лежить в основі численних децентралізованих фінансових додатків (DeFi), зокрема, децентралізованих бірж, кредитних платформ та систем управління активами. Також Solidity використовується для автоматизації та оптимізації логістичних процесів у ланцюгах поставок, де важливі прозорість та надійність операцій. Ця мова програмування є ключовою для створення систем децентралізованого голосування та управління, що забезпечують безпеку та незмінність даних, надаючи значні переваги для суспільних та корпоративних організацій.

У результаті, Solidity відіграє критичну роль у розвитку і вдосконаленні технологій блокчейну, надаючи розробникам потужні інструменти для створення складних і безпечних смарт-контрактів, які є фундаментом для інноваційних додатків і служб.

Наведемо приклад простого смарт-контракту на Solidity, який реалізує найпростішу функціональність – зберігання та отримання числа:

```solidity
pragma solidity ^0.8.0;
contract SimpleStorage {
    uint storedData;
    function set(uint x) public {
        storedData = x;
    }
    function get() public view returns (uint) {
```

```
        return storedData;
    }
}
```

Пояснення коду:

1. Встановлення версії компілятора:

   pragma solidity ^0.8.0;

Ця команда визначає, що для компіляції контракту необхідно використовувати компілятор Solidity версії 0.8.0 або новішу.

2. Створення контракту:

   solidity

   contract SimpleStorage {

З цього моменту починається визначення контракту під назвою SimpleStorage.

3. Змінна для зберігання даних:

   solidity

   uint storedData;

Контракт містить змінну storedData, призначену для зберігання числа типу uint (це скорочення від unsigned integer, тобто беззнакове ціле число).

4. Функція для встановлення значення:

   solidity

   function set(uint x) public {

   storedData = x;

   }

Ця функція дозволяє записати в storedData число, яке передається їй у параметрі x. Оскільки вона має модифікатор public, її можна викликати ззовні контракту.

5. Функція для отримання значення:

   solidity

   function get() public view returns (uint) {

```
return storedData;
}
```

Функція get повертає значення змінної storedData. view в її описі означає, що вона лише читає дані і не модифікує стан контракту, тому виклик такої функції не витрачає газ, якщо не враховувати вартість транзакції.

1.1.10 Криптовалютні гаманці: види та принципи роботи

Криптовалютний гаманець є ключовим інструментом для забезпечення безпеки цифрових активів, оскільки він управляє ключами, що дають доступ до криптовалюти, збереженої у блокчейні. Використання пари ключів — відкритого для отримання коштів та закритого для авторизації транзакцій — забезпечує значний рівень захисту. Система секретних фраз відновлення, регульована специфікацією BIP-39, додатково підсилює безпеку, надаючи користувачам надійний спосіб відновлення доступу до коштів у випадку втрати доступу до пристроїв.

Щодо ризиків взлому криптовалютного гаманця методом "грубої сили", то вони є мінімальними, зважаючи на кількість можливих комбінацій у секретній фразі. Якщо розглядати 12-слівну фразу зі списку BIP-39, де кожне слово має 2048 можливостей, загальна кількість комбінацій розраховується за формулою:

Кількість комбінацій = 2048^{12}

Це призводить до приблизно $5.44451787 \times 10^{39}$ комбінацій, що є неймовірно великим числом. Навіть якщо зловмисник намагатиметься перебирати одну комбінацію в секунду, йому знадобилося б більше мільярдів років, щоб перебрати всі можливі комбінації. Розрахунок часу, необхідного для перебору:

Час на перебір = $(2048^{12}) / (60 * 60 * 24 * 365.25)$ років

Цей розрахунок підкреслює фактичну нездійсненність взлому гаманця методом "грубої сили". Щоб ілюструвати шанси на взлом, припустимо, що атака здійснюється з швидкістю мільярд комбінацій в секунду. Навіть при таких умовах, шанси зламати гаманець за один день є просто несуттєвими:

Шанс взлому за день = $(86{,}400 * 10^{9}) / (2048^{12}) \approx 1.59 * 10^{-27}$

Генерація секретної фрази відновлення з використанням BIP-39

Для створення секретної фрази відновлення потрібно згенерувати від 128 до 256 біт ентропії. Залежно від довжини початкової ентропії, кількість слів у секретній фразі може коливатися від 12 до 24.

Розглянемо процес створення секретної фрази з 12 словами. Спочатку генерується 128 біт ентропії:

Таблиця 1.3. початкова ентропія для генерації секретної фрази відновлення

11111011 00010101 11111100 00011110 01000100 00011011 10110100 00110001 01000000 01111010 01010111 11101011 10000111 11111010 00011111 11011110

Після цього використовується хеш SHA-256 для отримання перших бітів, які слугують контрольною сумою. Додавання цієї контрольної суми до ентропії призводить до формування 132 біт:

Таблиця 1.4. бітова послідовність та відповідні їм слова за стандартом BIP-39

11111011000	10101111111	00000111100	10001000001	10111011010
2008	1407	60	1089	1498
00011000101	00000001111	01001010111	11101011100	00111111110
197	15	599	1884	510
10000111111	10111100100			
1087	1508			

Кожна з цих послідовностей біт переводиться у десяткове число, що використовується для вибору відповідного слова зі словника BIP-39. Словник містить 2048 слів, доступних на різних мовах, зокрема англійською, китайською та іспанською. Вибір кожного слова залежить від відповідного числа в десятковій системі. Наприклад, для англійської версії словник виглядає так:

Таблиця 1.5.секретна фраза відновлення зі словником BIP-39

2008 1407 60 1089 1498 197 15

wild quiz always market robust board acid

599 1884 510 1087 1508

Enough twist divert margin route

Види криптовалютних гаманців:

1. Апаратні гаманці: Це фізичні пристрої, які зберігають ключі в автономному режимі. Вони вважаються найбезпечнішим видом гаманця.
2. Програмні гаманці: Це програми, які ви інсталюєте на свій комп'ютер або телефон. Вони бувають різних типів, включаючи мобільні гаманці, веб-гаманці та десктопні гаманці.
3. Кастодіальні гаманці: Це гаманці, у яких ваші ключі зберігаються стороннім сервісом. Це зручно, але менш безпечно, ніж зберігання ключів самостійно.
4. Некастодіальні гаманці: Це гаманці, у яких ви зберігаєте свої ключі. Це безпечніше, але вимагає від вас більше відповідальності.

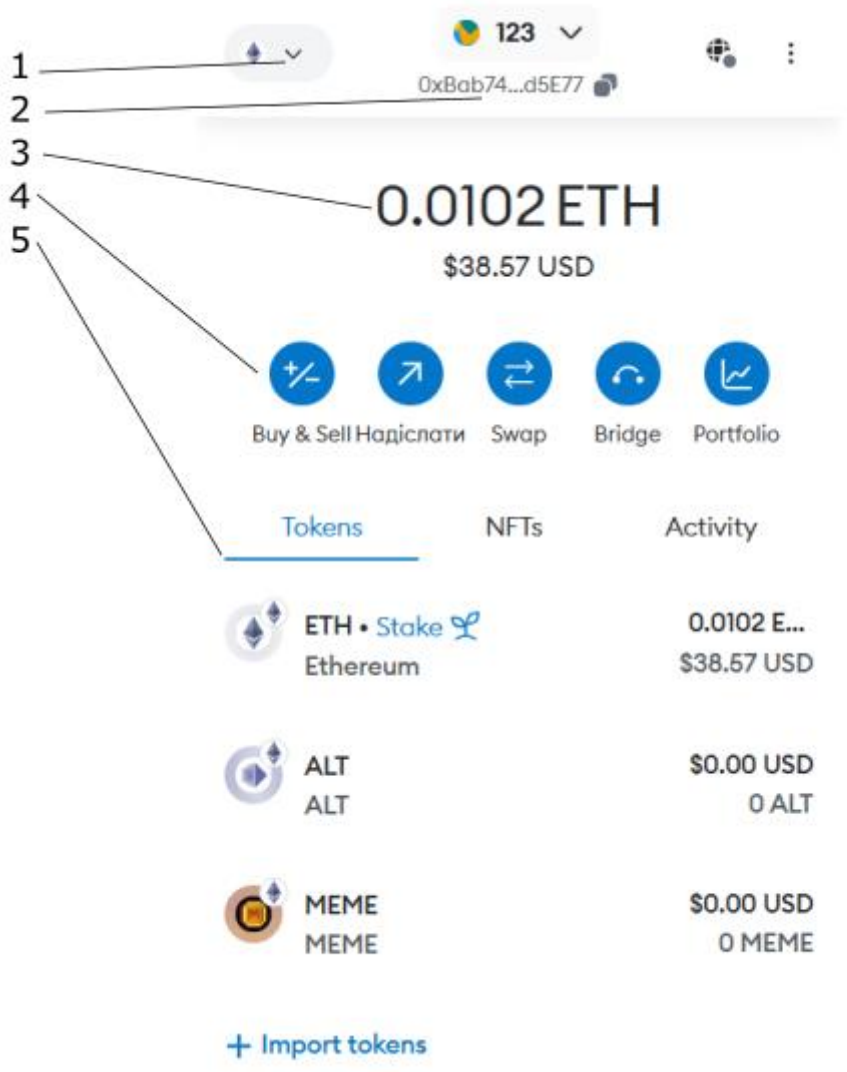


Рисунок 1.14. інтерфейс некастодіального криптовалютного гаманця Metamask.

Відповідно до рисунку 1.14:

1. Іконка мережі: Значок, що вказує на поточну мережу, до якої підключений гаманець (наприклад, Ethereum Mainnet).
2. Іконка гаманця та адреса гаманця: Значок гаманця, що показує доступ до облікового запису користувача, та унікальний ідентифікатор гаманця в блокчейні, який можна використовувати для надсилання та отримання криптовалют.
3. Баланс ETH: Поточний баланс ефіру (ETH) у гаманці та його еквівалент у доларах США (USD).

4. Меню дій: Кнопки для виконання різних дій, таких як купівля та продаж, надсилання, обмін, міст (Bridge) для переказу активів між мережами та портфоліо.
5. Баланс токенів: Інформація про наявні токени у гаманці, їхні назви та поточні баланси у відповідних одиницях та доларах США (USD).

1.2 Основи взаємодії з криптовалютою та децентралізованими додатками

1.2.1 Огляд децентралізованих додатків

Uniswap представляє собою протокол для децентралізованого обміну (DEX), який революціонізує спосіб торгівлі криптовалютами. Він дозволяє користувачам безпосередньо обмінюватися токенами без потреби залучати централізовані біржі як посередників, забезпечуючи високий рівень безпеки та прозорості. Основа роботи Uniswap полягає у використанні автоматизованих пулів ліквідності та розумних контрактів, що управляють обміном. Ці пули функціонують на принципах саморегулювання, де курс обміну токенів автоматично адаптується залежно від змін у попиті та пропозиції. Кожен пул складається з пар токенів, і користувачі можуть виконувати обмін прямо в межах цих пар, що значно спрощує процес торгівлі в децентралізованому середовищі.

Основні компоненти Uniswap:

1. Пули ліквідності: Кожен пул ліквідності містить два токена, наприклад, ETH та DAI, і користувачі можуть обмінювати їх один на одного. Пули ліквідності створюються розумними контрактами і може бути використані обміну токенів;
2. Курси обміну: Курси обміну в Uniswap визначаються принципом автоматичного регулювання на основі попиту та пропозиції всередині пулу ліквідності. Чим більше токенів одного виду торгуються за іншою, тим вищою стає ціна цього токена;
3. Комісії: Uniswap стягує комісії за кожну транзакцію, яка йде на винагороду ліквідаторам та забезпечує інцентиви для учасників мережі;

4. Розумні контракти: Вся функціональність Uniswap реалізована через розумні контракти на блокчейні Ethereum, що забезпечує прозорість, надійність та безпеку торгівлі.

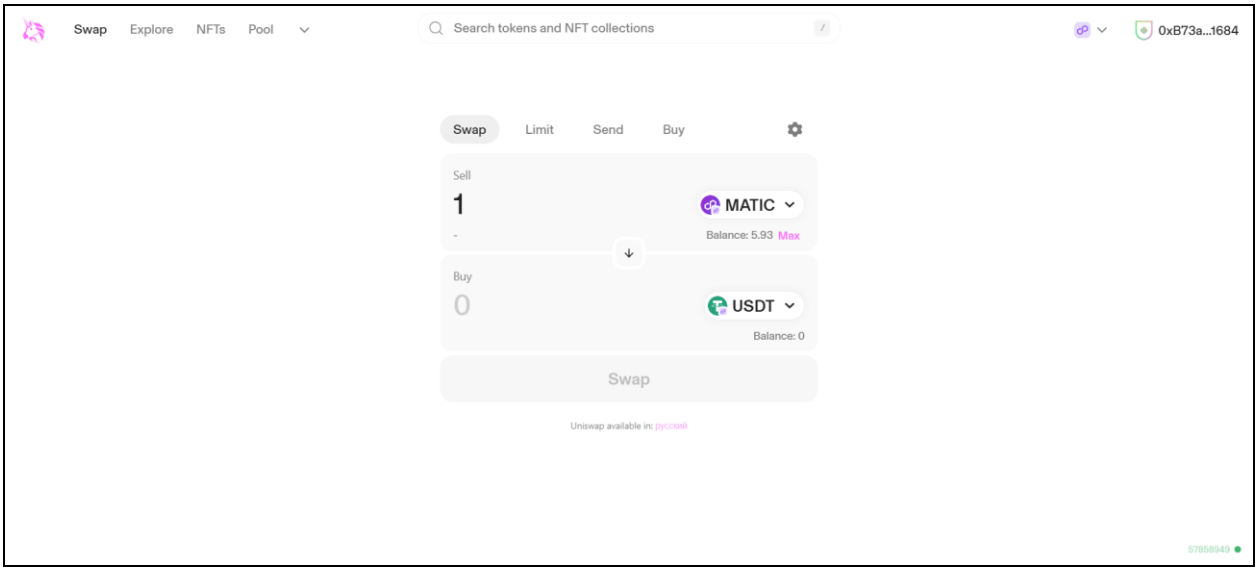


Рисунок 1.15. Інтерфейс для обміну криптовалютами на платформі Uniswap.

Aave є інноваційним децентралізованим протоколом для кредитування та позичання цифрових активів, розробленим на базі блокчейну Ethereum. Цей протокол надає користувачам можливість виконувати такі фінансові операції, як позики та запозичення, без потреби звертатися до традиційних банків або кредитних агенцій. Завдяки використанню розумних контрактів, Aave гарантує високий рівень безпеки та прозорості, одночасно автоматизуючи всі процеси, що робить фінансові операції більш ефективними та доступними для широкої аудиторії.

Основні функції Aave

1. Запозичення (Borrowing): Користувачі можуть зайняти цифрові активи, надавши інші активи як заставу. Позичальники можуть отримати доступ до кредитів без необхідності проходження кредитного скорингу або надання особистих даних.
2. Депозити: Користувачі можуть внести цифрові активи в депозит, щоб отримувати відсотки за своїм коштом. Депозитори можуть заробляти відсотки на своїх активах, що використовуються для забезпечення ліквідності у системі.

3. Ліквідність (Liquidity): Aave забезпечує ліквідність для торгівлі та запозичення цифрових активів, дозволяючи користувачам обмінювати свої активи безпосередньо один з одним.
4. Процентні ставки (Interest Rates): Протокол автоматично регулює процентні ставки за кредитами та депозитами на основі попиту та пропозиції в системі.
5. Розумні контракти (Smart Contracts): Вся функціональність Aave реалізується через розумні контракти на блокчейні Ethereum, що забезпечує безпеку та безперервне функціонування протоколу.

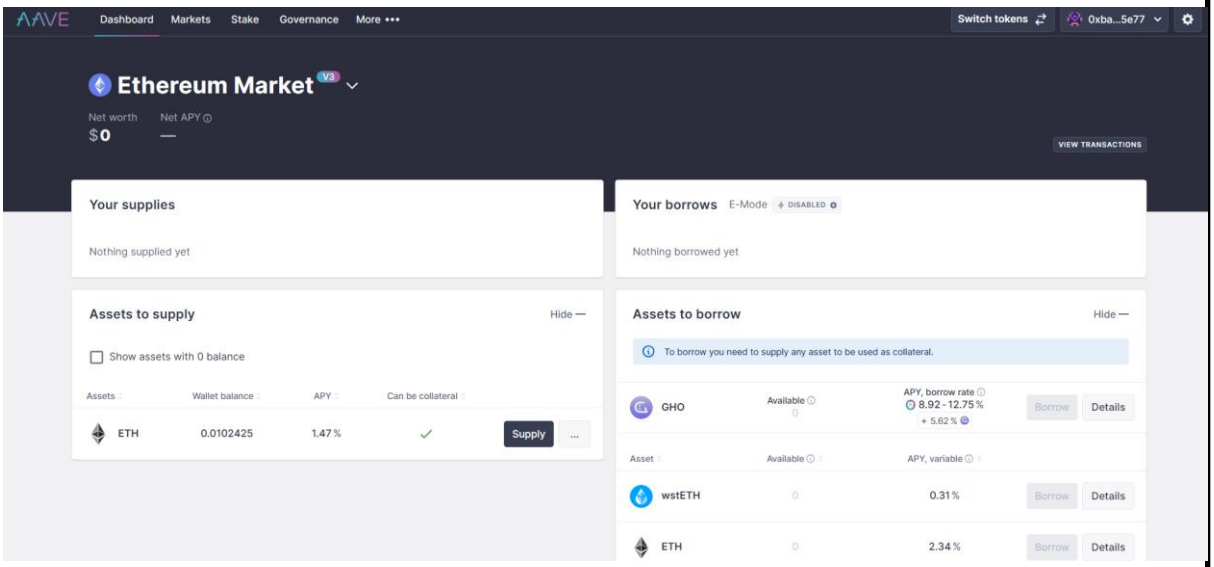


Рисунок 1.16. Інтерфейс користувача ринку Ethereum на платформі Aave для позичання та кредитування.

Aragon представляє собою децентралізовану платформу, розроблену на основі блокчейну Ethereum, яка спрощує процес створення та керування організаціями. Платформа забезпечує необхідні інструменти та інфраструктуру для формування децентралізованих автономних організацій (DAO). Ці організації використовуються для ефективного управління колективами людей, активами та іншими ресурсами без втручання центральних органів, забезпечуючи повну автономію у прийнятті рішень та управлінні.

Ключові аспекти та функції Aragon

1. Створення DAO: Aragon дозволяє користувачам створювати нові децентралізовані організації, які називаються DAO. Користувачі

можуть визначати правила управління, розподіл активів та принципи прийняття рішень усередині DAO.

2. Управління голосуванням: Платформа надає інструменти для проведення голосувань та прийняття рішень у рамках DAO. Учасники можуть голосувати з різних питань, таких як зміна правил, розподіл фондів або найм персоналу.
3. Інтеграція з Ethereum: Aragon базується на блокчейні Ethereum, що забезпечує безпеку та прозорість операцій. Смарт-контракти Ethereum використовуються для автоматизації процесів керування та голосування всередині DAO.
4. Розширюваність та гнучкість: Платформа розроблена з урахуванням гнучкості та розширюваності, що дозволяє користувачам налаштовувати правила та функціональність своїх DAO відповідно до їх унікальних потреб та вимог.
5. Інтерфейс та користувальницький досвід: Aragon надає зручний та інтуїтивно зрозумілий інтерфейс для управління та взаємодії з DAO. Це включає веб-інтерфейс, мобільні додатки та інші інструменти для управління організацією.

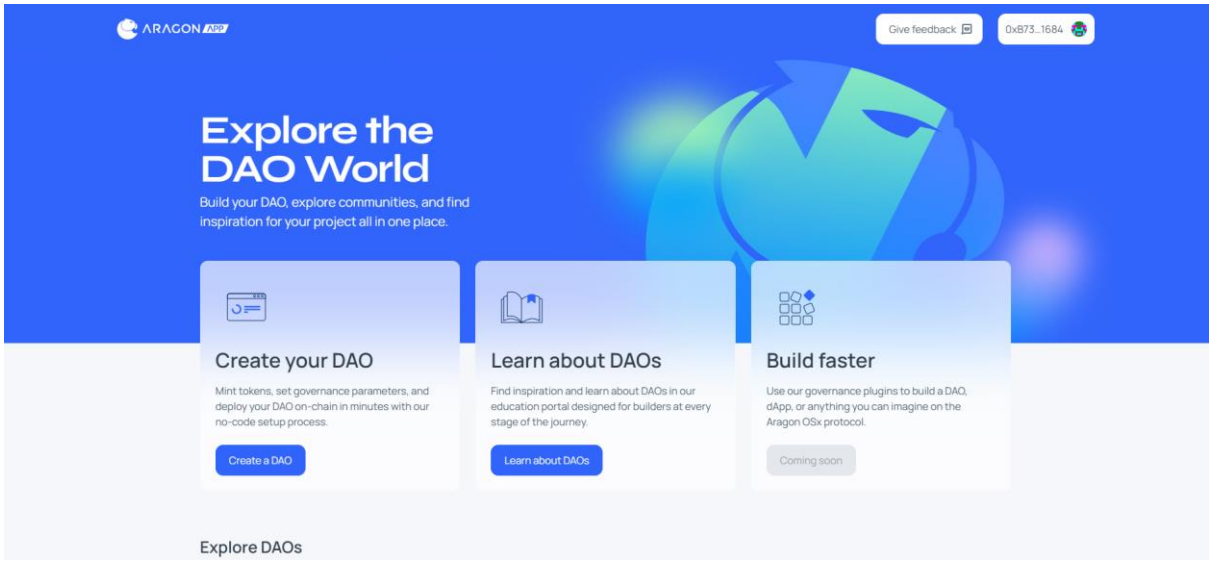


Рисунок 1.17. Інтерфейс веб-сайту Aragon для ознайомлення та створення DAO.

1.2.2 Основи RCP: Протокол віддаленого виклику процедур

Remote Procedure Call (RCP) або віддалений виклик процедур - це протокол, що дозволяє програмам виконувати функції на віддаленому сервері, якби вони були локальними викликами. У контексті криптовалют, RCP часто використовується для взаємодії з блокчейном, виконання транзакцій, отримання даних про блоки та інші операції.

RCP в криптовалюті є набір методів та інтерфейсів, які дозволяють програмам взаємодіяти з блокчейном через мережу. Протокол RCP працює за принципом клієнт-серверної архітектури, де клієнт надсилає запит серверу, який виконує необхідну операцію та повертає результат.

Основні компоненти RCP включають:

1. Клієнт: програма або скрипт, який ініціює запит.
2. Сервер: Вузол у мережі блокчейн, який приймає запити та виконує їх.
3. Процедури: Визначені функції, які можуть бути викликані віддалено, такі як відправка транзакції, отримання інформації про блок і т.д.

Використання RCP для взаємодії з блокчейном: Автоматизація, Скриптинг та Інтеграція:

- Доступ до Блокчейну: RCP дозволяє програмам та користувачам взаємодіяти з блокчейном без необхідності запуску власного вузла. Це спрощує процес доступу до даних та виконання операцій у блокчейні;
- Автоматизація та Скриптинг: За допомогою RCP можна автоматизувати різні процеси, такі як створення та надсилання транзакцій, моніторинг мережі, управління гаманцями тощо. Це особливо корисно для трейдерів, розробників та компаній, які працюють із криптовалютами;
- Інтеграція з Додатками: RCP використовується для інтеграції блокчейн-функціоналу до веб-додатків, мобільних додатків та інших програмних систем. Це дозволяє користувачам взаємодіяти із блокчейном через знайомий інтерфейс;

– Підвищення ефективності: Використання RCP дозволяє знизити навантаження на локальні ресурси, оскільки важкі операції виконуються на віддалених серверах. Це особливо актуально для роботи з великими даними та виконання складних обчислень.

Для отримання RCP можна звернутися до сайту chainlist.org(відповідно до рисунку 2.4), де представлений великий перелік публічних RCP для різноманітних блокчейнів.

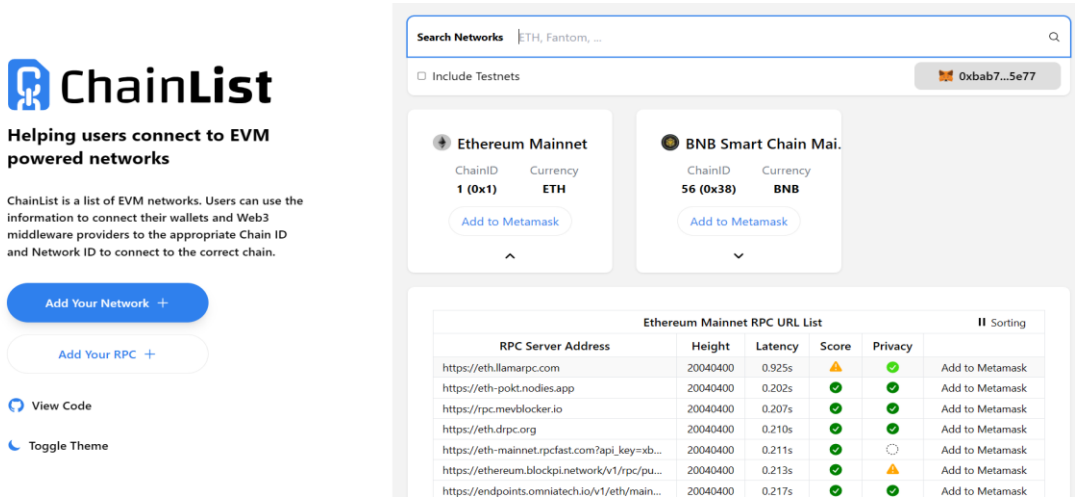


Рисунок 1.18. інтерфейс веб-сайту ChainList, який надає перелік публічних RPC-серверів для блокчейн-мереж

Приклад коду взаємодії з блокчейном за допомогою RCP: Від імпорту до відправлення транзакцій:

```
from web3 import Web3

rcp_url = " https://eth.llamarpc.com"
web3 = Web3(Web3.HTTPProvider(rcp_url))
transaction = {
    'to': 'RECEIVER_ADDRESS',
    'value': web3.toWei(0.01, 'ether'),
    'gas': 2000000,
    'gasPrice': web3.toWei('50', 'gwei'),
    'nonce':
web3.eth.getTransactionCount('YOUR_WALLET_ADDRESS'),
```

```
}


signed_txn = web3.eth.account.signTransaction(transaction,
private_key='YOUR_PRIVATE_KEY')
tx_hash = web3.eth.sendRawTransaction(signed_txn.rawTransaction)
print(web3.toHex(tx_hash))
```

Пояснення коду:

Коли ми починаємо працювати з блокчейном, першим кроком є підключення потужної бібліотеки Web3. Ця бібліотека дозволяє нам безпосередньо взаємодіяти з блокчейном, використовуючи Python.

```
from web3 import Web3
```

Щоб почати, нам потрібно з'єднатися з Блокчейном. Завдяки Chainlist ми можемо легко отримати доступ до мережі через простий URL, що забезпечує швидке та надійне з'єднання

```
rcp_url = " https://eth.llamarpc.com"
web3 = Web3(Web3.HTTPProvider(rcp_url))
```

Наступним кроком є створення транзакції. Ми вказуємо, куди і скільки ефіру відправити, а також встановлюємо вартість газу та інші параметри для забезпечення успішної обробки.

```
transaction = {
   'to': 'RECEIVER_ADDRESS',
   'value': web3.toWei(0.01, 'ether'),
   'gas': 2000000,
   'gasPrice': web3.toWei('50', 'gwei'),
   'nonce':
web3.eth.getTransactionCount('YOUR_WALLET_ADDRESS'),
}
```

Перед тим як відправити транзакцію, ми підписуємо її нашим приватним ключем для забезпечення її автентичності. Після підпису, транзакція

відправляється в блокчейн, а її хеш може бути використаний для відстеження статусу транзакції.

```
signed_txn = web3.eth.account.signTransaction(transaction,
private_key='YOUR_PRIVATE_KEY')
tx_hash = web3.eth.sendRawTransaction(signed_txn.rawTransaction)
print(web3.toHex(tx_hash))
```

1.2.3 Виклики зручності користування криптовалют

Криптовалюти, незважаючи на їх популярність і потенціал для революціонізування фінансових систем, стикаються з кількома загальними проблемами. Однією з найбільш значущих проблем є їх незручність для звичайних користувачів (не зручні для користувачів). Ось кілька аспектів, які складають цю проблему:

1. Складність технологій: Розуміння основ криптовалют і блокчейн-технологій вимагає певних технічних знань. Для багатьох користувачів такі поняття, як "блокчейн", "хеш", "смарт-контракт", "майнінг криптовалют" та інші, можуть бути складними і заплутаними.
2. Труднощі з використанням гаманців: Криптовалютні гаманці вимагають уважного поводження з приватними ключами і seed-фразами. Втрата цих даних може призвести до втрати всіх засобів, і немає можливості їх відновити. Для багатьох користувачів це створює відчуття ненадійності і страху втратити свої активи.
3. Користувацькі інтерфейси: Багато криптовалютних платформ і бірж мають складні інтерфейси, які можуть бути важкими для розуміння і навігації, особливо для новачків. Це може відлякувати нових користувачів і ускладнювати масове прийняття криптовалют.
4. Слабка інтеграція з традиційними фінансовими системами: Використання криптовалют для повсякденних платежів і переказів часто пов'язане з необхідністю конвертації у фіатні гроші, що може бути незручним і дорогим через комісії та волатильність курсів.

5. Відсутність підтримки та сервісів: У разі проблем або питань, у користувачів криптовалют часто немає доступу до служби підтримки, до якої вони звикли в традиційних фінансових установах. Це може створити додаткові бар'єри для нових користувачів.
6. Безпека і шахрайство: Криптовалютні транзакції незворотні, що робить користувачів вразливими для шахрайства і помилок. Відсутність центрального регулюючого органу також означає, що користувачі несуть повну відповідальність за свою безпеку.
7. Рішення цих проблем вимагає поліпшення освітніх ресурсів, розробки більш інтуїтивних і безпечних додатків, а також поліпшення інтеграції криптовалют з існуючими фінансовими системами. Тільки тоді криптовалюти зможуть стати по-справжньому доступними і зручними для широкого кола користувачів.

1.2.4 Проблема ведення декількох гаманців і для чого це потрібно

Стратегії забезпечення безпеки:

- Диверсифікація ризиків: Застосування правила "не класти всі яйця в один кошик" є важливим у кібербезпеці. Використання декількох гаманців допомагає розподілити активи, тим самим знижуючи ризик втрати усіх коштів у разі атаки на один з гаманців. Наприклад, якщо доступ до одного гаманця буде втрачено, інші залишаться недоторканими.
- Зберігання резервів: Використання окремих гаманців для тривалого зберігання коштів зменшує ризик їх втрати. Так звані "холодні" гаманці, які не підключені до мережі, важко зламати, що робить їх відмінним рішенням для зберігання значних сум.
- Мінімізація збитків: Розподіл активів між кількома гаманцями допомагає обмежити збитки в разі втрати доступу до одного з них. Таким чином, втрата ключів до одного гаманця не призведе до втрати усіх активів.

Фінансові переваги:

- Оптимізація комісій: Використання різних гаманців дозволяє скористатися можливістю оптимізації комісій за транзакції. Користувачі можуть використовувати гаманці з нижчими комісіями для частих транзакцій та більш захищені гаманці для довгострокового зберігання.
- Хеджування ризиків: Різноманітність гаманців для різних типів активів дозволяє ефективніше управляти фінансовими ризиками. Наприклад, частина коштів може бути інвестована у стабільні криптовалюти, тоді як інша – в активи з вищим потенціалом прибутку.
- Контроль витрат: Множинні гаманці сприяють чіткому обліку та контролю витрат. Користувачі можуть використовувати окремі гаманці для інвестицій, щоденних витрат, заощаджень, що дозволяє краще планувати і відстежувати фінансові потоки.

Виклики множинного управління гаманцями:

- Складність управління: Чим більше гаманців у користувача, тим складніше їх контролювати. Необхідність запам'ятовувати ключі від кожного гаманця збільшує ризик втрати доступу до коштів.
- Збільшені вимоги до безпеки: Кожен гаманець потребує окремих заходів безпеки, таких як унікальні паролі та двофакторна аутентифікація, що збільшує загальну складність.
- Труднощі інтеграції: Використання багатьох гаманців може ускладнити інтеграцію з сервісами і біржами, а також переказ коштів між ними або використання для оплати товарів і послуг.

1.3 Аналіз скрипта для відправки даних у блокчейн

1.3.1 Автоматизація блокчейн-транзакцій: оптимізація взаємодії та управління ресурсами

Цей скрипт створений для того, щоб автоматизувати процес відправлення транзакцій у мережі блокчейну, використовуючи попередньо заданий набір

приватних ключів. Це значно пришвидшує взаємодію з блокчейном та робить її більш зручною. Скрипт не просто ініціює підключення до різних блокчейн-мереж, а й розумно управляє транзакціями, адаптуючись до коливань ціни на газ, що є важливим для ефективності витрат. Крім того, він використовує асинхронні методи для оптимізації доступу до ресурсів, забезпечуючи плавне та ефективне виконання завдань у конкурентному середовищі. Такий підхід дозволяє не тільки зменшити час на обробку транзакцій, але й збільшує загальну продуктивність взаємодії з блокчейн-технологіями.

1.3.2 Загальна структура кода

Рисунок 1.19. Схема процесу відправки транзакцій за допомогою скрипта для автоматизації.

Скрипт використовує бібліотеку asyncio для асинхронного виконання завдань, Web3 для взаємодії з блокчейном Ethereum та loguru для логування. Основні компоненти включають функції worker, main та допоміжні модулі для конфігурації і роботи з транзакціями.

1. Функція worker:

За допомогою функції worker відбувається виконання транзакцій для кожного приватного ключа. Алгоритм роботи наступний:

Витяг приватного ключа з черги.

Створення об'єкта Web3Wrapper, який спрощує взаємодію з блокчейном.

Перевірка поточної ціни газу. Якщо ціна занадто висока, виконання завдання відкладається.

Відправка транзакції з заданими параметрами.

Приклад коду:

```
async def worker(q: asyncio.Queue, web3: Web3, config: Config, locker: asyncio.Lock) -> None:
    while not q.empty():
        if locker.locked():
        await asyncio.sleep(1)
        continue
        private_key = await q.get()
        try:
        w3 = Web3Wrapper(web3=web3, private_key=private_key)
        except Exception as err:
        logger.error(f"Error while creating Web3 object: {err}")
        continue
        sleep_time = round(uniform(*config.SLEEP_BETWEEN_ACCOUNT_WORK), 3)
```

```
            logger.info(f"[{w3.account_address}] Sleep for {sleep_time} seconds")
            await asyncio.sleep(sleep_time)
            if (gas_price := await w3.web3.eth.gas_price) > config.GAS_TARGET *
1e9:
             logger.info(f"[{w3.account_address}] Gas price is too high ({gas_price /
1e9} gwei), skipping")
            q.put_nowait(private_key)
            continue
            try:
              tx_hash = await w3.estimate_and_send_transaction(
                    tx_params={
                    "to": config.CONTRACT,
                    "data": config.INPUT_DATA,
                    "value": config.VALUE,
                    **(await w3.eip1559_gas_price),
                    }
            )
            except Exception as err:
             logger.error(f"[w3.account_address}] Error while sending transaction:
{err}")
            continue
          logger.success(f"[{w3.account_address}]  Transaction  sent:
{tx_hash.hex()}")
```

2. Функція main:

Функція main ініціалізує конфігурацію і логування, встановлює підключення до блокчейну Ethereum, завантажує приватні ключі з файлу і запускає асинхронні робочі завдання для обробки транзакцій.

Приклад коду:

```
async def main() -> None:
config = load_config()
```

```
web3 = Web3(Web3.AsyncHTTPProvider(config.RPC), middlewares=[], modules={"eth": (AsyncEth,)})
locker = asyncio.Lock()
q = asyncio.Queue()
   asyncio.create_task(gas_locker(locker, web3, config.GAS_TARGET))
with open("./src/private_keys.txt") as f:
        private_keys = f.read().splitlines()
if not private_keys:
        logger.error("No private keys provided, exiting")
        sys.exit(1)
[q.put_nowait(pk) for pk in private_keys]


workers = [asyncio.create_task(worker(q, web3, config, locker)) for _ in range(config.WORKERS_COUNT)]
await asyncio.gather(*workers)
```

3. Функція load_config:

Функція load_config() завантажує конфігураційні параметри з файлу config.yaml і створює об'єкт Config, який містить всі необхідні параметри для виконання основного скрипта.

Приклад коду:

```
def load_config() -> Config:
with open("./src/config.yaml", "r") as f:
        config = yaml.safe_load(f)
return Config.parse_obj(config)
```

4. Клас Web3Wrapper

Клас Web3Wrapper надає зручні методи для взаємодії з мережею, використовуючи приватний ключ для підпису транзакцій.

Основні методи:

__init__: Ініціалізує об'єкт Web3Wrapper з веб3 і приватним ключем.

estimate_and_send_transaction: Оцінює газ і відправляє транзакцію.

eip1559_gas_price: Повертає параметри газу за EIP-1559.

Приклад коду:

```
class Web3Wrapper:
    def __init__(self, web3: Web3, private_key: str) -> None:
        self.web3 = web3
        self.__account = Account.from_key(private_key)
        self.account_address = self.__account.address
    async def estimate_and_send_transaction(self, tx_params: TxParams, gas_multiplier: int | float = 1.05) -> HexBytes:
        if not self.web3.is_checksum_address(tx_params["to"]):
        tx_params["to"] = self.web3.to_checksum_address(tx_params["to"])

        if not tx_params.get("nonce"):
        tx_params["nonce"] = await self.web3.eth.get_transaction_count(self.account_address, "pending")
        if not tx_params.get("from"):
        tx_params["from"] = self.account_address
        if not tx_params.get("chainId"):
        tx_params["chainId"] = await self.web3.eth.chain_id
        gas = await self.web3.eth.estimate_gas(tx_params)
        tx_params["gas"] = int(gas * gas_multiplier)
        signed_tx = self.__account.sign_transaction(tx_params)
        return await self.web3.eth.send_raw_transaction(signed_tx.rawTransaction)
    async def eip1559_gas_price(self) -> dict[str, int]:
        base_fee = await self.web3.eth.gas_price
        max_priority_fee = await self.web3.eth.max_priority_fee
        return {
        "maxPriorityFeePerGas": max_priority_fee,
        "maxFeePerGas": (base_fee + max_priority_fee) * 2,
```

```
}
```

5. Функція gas_locker:

Функція gas_locker блокує виконання завдань при високій ціні газу. Вона використовує асинхронне блокування для запобігання багаторазовому виклику з різних корутин.

Приклад коду:

```
async def gas_locker(locker: asyncio.Lock, web3: Web3, gas_target: float | int) -> None:
    gas_target = gas_target * 1e9
    logger.info(f"Gas locker set to {gas_target / 1e9} gwei")
    await locker.acquire()

    while True:
        if (current_gas := await web3.eth.gas_price) > gas_target:
        if not locker.locked():
            await locker.acquire()
        logger.info(f"Current gas price: {current_gas / 1e9} gwei, waiting for {gas_target / 1e9} gwei")
        await asyncio.sleep(1)
        else:
        if locker.locked():
            locker.release()
            logger.info(f"Current gas price: {current_gas / 1e9} gwei. Proceeding.")
```

6. Функція configure_logger:

Функція configure_logger() налаштовує параметри логування, створюючи два логери: один для запису у файл, інший для виводу в консоль з кольоровим форматуванням.

Приклад коду:

```
def configure_logger() -> None:
```

```
    logger.remove()
    logger.add(
        "logs/{time:MM_D}/{time:HH_mm}.log",
          format="{time:HH:mm:ss} | {function}:{line} | {level} - {message}",
    )
    logger.add(
          sys.stdout,
        format="<level>{time:HH:mm:ss}</level> | <lk>{function}</lk>:<lk>{line}</lk> | <level>{level}</level> <magenta>{message}</magenta>",
          colorize=True,
    )
```

2. ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

Темою даного дипломного проекту є розробка скрипт для автоматизації транзакцій у блокчейн-мережах, який використовує заздалегідь задані приватні ключі, забезпечуючи швидкість та зручність взаємодії з блокчейном.

Ефективність програмного продукту визначається його якістю та ефективністю процесу розробки. Якість ПП визначається наступними складовими: з точки зору користувача; з позиції використання ресурсів; виконання вимог до програмного забезпечення. Оцінка якості програмного продукту включає визначення трудомісткості і вартості його створення.

Проведемо розрахунки визначення трудомісткості розробки даного програмного продукту.

2.2 Визначення трудомісткості розробки програмного забезпечення.

Тривалість розробки програмного продукту залежить від його обсягу, трудомісткості розробки, кваліфікації виконавців, а також планових термінів, визначених умовами ринку.

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт.

Таблиця 2.1. Каталог аналогів

Найменування ПП	**Обсяг функції ПП – V_o, усл. машинних командах.**
1. ПП СУБД	2500 – 9800
2. Комплексні системи ведення БД	950 – 7430
3. ПП оптимізації розрахунків	1300 – 4200

Методом структурної аналогії по відповідних каталогах аналогів програмного забезпечення визначаємо обсяг програмних засобів, у тисячах умовних машинних команд програми аналога. Для нашого варіанта виділено сірим кольором.

Вибравши аналог ПП, що містить Vo в умовних машинних командах, трудомісткості визначаємо на основі табл.2.2

Таблиця.2.2. Трудомісткість

Обсяг ПП, тис.умов.машинних команд	Норма часу, люд/год
1.00	229
2.00	244
3.00	262
4.00	283

На підставі отриманого значення, по довіднику, визначаємо укрупнену норму часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПП, тобто в умовах комп'ютера, Кк=0,7÷0,8): $T^a_p = 229$ (люд/годин).

Трудомісткість програмного продукту визначається по кожному етапу розробки окремо на підставі трудомісткості аналога з урахуванням складності розробки, ступеня новизни і ступеня використання в розробці стандартних модулів на підставі формул:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \quad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \quad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \quad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розробки (див. табл. 2.3.);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.4.);

Кт – поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.5.).

Таблиця 2.3. Значення питомих коефіцієнтів трудомісткості і-го етапу в загальній трудомісткості розробки ПП.

Код стадії	Ступінь новизни		
	А	Б	В
ТЗ (L_1)	0,15	0,12	0,12
ТП (L_2)	0,16	0,15	0,11
РП (L_3)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	Ступінь новизни	Значення $К_н$
А	Принципово нові ПО	1,75 – 1,2
Б	ПО – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПО маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.5. Значення коефіцієнта ступеня використання в розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПО типовими програмами, %	Значення $К_т$
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=Та*L_1*К_н=229×0,12×1,0 = 27,48 \quad (люд/годин) \qquad (2.1)$$

Трудомісткість розробки технічного проекту

$$Т_{тп}=Та*L_2*К_н=229×0,15×1,0 = 34,35 \quad (люд/годин) \qquad (2.2)$$

Трудомісткість розробки робочого проекту

$$Т_{рп}=Та*L_3*К_н*К_т=229×0,58×1,0×0,7 = 92,97 \quad (люд/годин) \qquad (2.3)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап:технічне завдання $N_{тз}$=2 (стр), розробка ТП $N_{тп}$=46(стр), розробка робочого проекту $N_{рп}$=11 (стр), пояснювальна записка відповідно $N_{пз}$=9 (стр)

Розрахунок зведений у таблицю 2.6

Таблиця 2.6. Розрахунок трудомісткості ПП

Найменування етапів	Розрахунок, години.		
1	Розробка ПП	Контроль керівника	Нормоконтроль
1.ТЗ	$Т_{Ртз}$=27,48	$Т_{кк}=0,7*N_{тз}$=1,4	$Т_{нк}=0,15*N_{тз}$=0,3
2.Розробка ТП	$Т_{Ртп}$=34,35	$Т_{кк}=0,7*N_{тп}$=32,2	$Т_{нк}=0,15*N_{тп}$=6,9

3.Розробка РП	$T_{Ррп}$=92,94	$T_{кк}$=0,7*$N_{рп}$=7,7	$T_{нк}$=0,15*$N_{рп}$=1,65
4.Розробка пояснювальної записки	$Tр_{пз}$=1,5*$N_{пз}$=13,5	$T_{кк}$=0,7*$N_{пз}$=6,3	$T_{нк}$=0,15*$N_{пз}$=1,35
Усього, в т.ч.:	T=226,07		
- на розробку	$\Sigma T_р$=168,27		
- контроль керівника		$\Sigma T_{ккмс}$=47,6	
- нормоконтроль			$\Sigma T_{нк}$=10,2

2.3 Розрахунок ціни програмного продукту.

У цьому розділі для визначення ціни розраховуємо основну заробітну плату виконавців, матеріальні витрати, вартість машино – години і витрати на розробку ПО. Розрахунок основної заробітної плати виконавців приведений у таблиці 2.7. Відповідно до статті 8 «Закону про Державний бюджет України на 2024» встановлено мінімальну заробітну плату у місячному розмірі з 1 квітня 2024 року - 8000 гривень; мінімальну погодинну тарифну ставку – 46,00 грн.

Таблиця 2.7. Розрахунок основної заробітної плати виконавців

Найменування робіт	Трудомісткість робіт, роб.години	Годинна тарифна ставка,грн..	Розрахунок, грн.
1.Розробка ПП	$\Sigma T_р$=168,27	46	7740,4
2.Контроль керівника	$\Sigma T_{кк}$=47,6	46	2189,6
3.Нормоконтроль	$\Sigma T_{кк}$=10,2	46	469,2
Усього ($З_о$)	-	-	Σ Зо= 10399,2

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо в таблицю 2.8

Таблиця 2.8. Розрахунок матеріальних витрат на розробку ПЗ

Найменування матеріальних витрат	Тип, модель	Кількість, шт	Ціна одиниці, грн.	Вартість, грн.

Папір	Лист	90	5	450
Разом	-	-	-	$В_{м\,і}$ =450
Транспортно – заготівельні Витрати 10%				$Втр\_з = 0{,}1 \times Вм1 = 45$
Усього				$В_м = В_{мі} + В_{тр\_з} = 495$

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.9.

Таблиця 2.9. Розрахунок статей витрат планової собівартості

Стаття витрат	Значення, грн.	Формула розрахунку
1. Матеріали	495	$В_м$ (див. табл. 2.8)
2. Основна заробітна плата	10399,2	$З_о$ (див. табл. 2.7)
3.Додаткова заробітна плата	1039,92	$Зд = 0{,}1 \times Зо = 0{,}1 \times 9662{,}2$
4.Відрахування до єдиного фонду соціального внеску	2516,6	$Вє.с.в. = 0{,}22 \times (Зо + Зд) =$ $0{,}22 \times (10399{,}2 + 1039{,}92)$
5. Накладні витрати	6239,52	$Внак. = 0{,}6 \times Зо = 0{,}6 \times 10399{,}2$
6. Повна собівартість	20690,24	$С_{пов} = В_м + З_о + З_д + Вє.с.в. + Внак.$

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$П = (С_{пов} * Р)/100 = (20690{,}24 * 10)/100 = 2069{,}02$ грн (2.4)

Де р – плановий рівень рентабельності (10-20%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$Ц_о = С_п + П = 20690{,}24 + 2069{,}02 = 22759{,}26$грн (2.5)

Виходячи з отриманих даних, ціна реалізації розробленого програмного продукту на основі наступної формули, становитиме:

$Ц_р = Ц_о + ПДВ = 22759{,}26 + 22759{,}26 * 0{,}2 = 27311{,}11$грн (2.6)

3 ОХОРОНА ПРАЦІ

Важливість безпеки праці і життєдіяльності особливо актуальна в контексті розвитку новітніх технологій, таких як блокчейн. Розробка і впровадження блокчейн-систем вимагає високої уваги до деталей безпеки, не тільки з точки зору програмного забезпечення, але й у фізичному середовищі розробників і користувачів цих систем.

3.1 Аналіз потенційних небезпечних і шкідливих факторів

Аналіз умов праці розробників блокчейн-технологій повинен включати оцінку ризиків, пов'язаних з тривалим сидінням за комп'ютером, взаємодією з серверним обладнанням та використанням великої кількості електронних пристроїв. До небезпечних факторів належать високий рівень шуму від серверів, ризик електричного удару, та психологічний стрес від роботи в режимі високої відповідальності і напруги.

3.2 Гігієнічні вимоги до виробничого середовища

Ефективність роботи та здоров'я працівників, які займаються розробкою та підтримкою блокчейн-систем, безпосередньо залежать від умов їх праці.

Оптимальне виробниче середовище повинне включати наступні аспекти:

- Ергономіка робочого місця: Належне облаштування робочих місць є ключовим для запобігання професійних захворювань. Столи мають бути оснащені регульованими по висоті поверхнями, щоб користувачі могли змінювати своє робоче положення від сидячого до стоячого. Крісла повинні мати підтримку спини, регульовані підлокітники та можливість налаштування кута нахилу.
- Мікроклімат: Оптимальні, допустимі й шкідливі норми температури, відносної вологості і швидкості руху повітря для виробничих приміщень та відкритих територій у спекотну і холодну пору року наведені в ДСН 3.3.6 042-99 «Санітарні норми мікроклімату виробничих приміщень». Робоче приміщення повинно мати ефективну систему вентиляції, яка забезпечує постійний притік свіжого повітря. Оптимальна температура

повітря становить 22-25°С, відносна вологість повинна бути на рівні 40-60%. Регулярне провітрювання приміщення допоможе уникнути накопичення шкідливих речовин та забезпечить комфортні умови для роботи.

Таблиця 1. Норми мікроклімату для приміщень з ПК

Пора року	Категорія робіт	Температура повітря, °С, не більше	Відносна вологість повітря, %	Швидкість руху повітря, м/с
Холодна	легка – 1а	22-24	40-60	0,1
	легка – 1б	21-23	40-60	0,1
Тепла	легка – 1а	23-25	40-60	0,1
	легка – 1б	22-24	40-60	0,2

- Вентиляція: Важливість адекватної вентиляції неможливо переоцінити, особливо в приміщеннях, де постійно працює велика кількість електроніки та комп'ютерного обладнання. Системи вентиляції, належно розроблені та інтегровані, відіграють ключову роль у забезпеченні здоров'я та комфорту персоналу. Вони повинні не тільки забезпечувати регулярний обмін повітря, але й оптимізувати внутрішнє повітряне середовище, видаляючи забруднене повітря, що може містити шкідливі гази, алергени, та частинки пилу, та подаючи чисте, свіже повітря ззовні. Це особливо актуально в офісах та лабораторіях, де недостатнє вентилювання може призвести до зниження продуктивності та підвищення рівня втоми серед працівників. Також, ефективна вентиляційна система може допомогти запобігти накопиченню тепла від електроніки, зберігаючи оптимальний баланс температури та вологості, що в свою чергу, сприяє довговічності обладнання та комфортним умовам праці.Рівні позитивних і негативних іонів у повітрі приміщень з ПК мають відповідати санітарно-гігієнічним нормам №2152–80 (табл. 2).

Таблиця 2. Рівні іонізації повітря приміщень при роботі на ПК

Рівні	Кількість іонів в 1 $см^3$ повітря	
	n+	n -
Мінімально необхідні	400	600
Оптимальні	1500-3000	3000-5000
Максимально допустимі	50000	50000

- Освітлення: Дослідження наукових публікацій показали, що недостатнє або неправильне освітлення може впливати на продуктивність та призводити до зорової втоми. Потрібно використовувати освітлювальні прилади, які мінімізують блиск і відблиски на моніторах. Також важливо забезпечувати достатнє природне освітлення.
- Психологічний комфорт: Робоче місце має сприяти психологічному комфорту. Це включає в себе забезпечення приватності, коли це необхідно, і можливості для соціальної взаємодії. Кольорове оформлення, рослини та вільний доступ до зон відпочинку можуть значно покращити загальний стан здоров'я та благополуччя працівників.
- Шумовий комфорт: Рівень шуму у виробничих приміщеннях, де розташовані сервери та інше обладнання, має бути мінімізований за допомогою звукоізоляційних матеріалів. Це допоможе зменшити ризик стресу та підвищити концентрацію працівників.

Забезпечення відповідності цих гігієнічних вимог є ключовим фактором у забезпеченні високої продуктивності та здоров'я працівників, зайнятих у сфері блокчейн-технологій.

3.3 Заходи з пожежної безпеки

Забезпечення пожежної безпеки – це один із важливих напрямків щодо охорони життя та здоров’я людей, національного багатства і навколишнього середовища. Основними причинами пожежі є: необережне поводження з вогнем, незадовільний стан електротехнічних установок і невиконання правил їх експлуатації, несправність виробничого обладнання і порушення режимів технологічних процесів, порушення правил пожежної безпеки.

До засобів гасіння пожежі відносяться внутрішні пожежні водопроводи (крани - ПК), вогнегасники (вуглекислотні та порошкові), сухий пісок тощо.

В будівлях пожежні крани встановлюють в коридорах, на майданчиках сходових кліток. Кожний пожежний кран укомплектований пожежним рукавом і розміщений у відповідних ящиках, які знаходяться на висоті 1,35 м від полу.

Для гасіння пожеж на початкових стадіях широко застосовуються вогнегасники. У виробничих приміщеннях це головним чином вуглекислотні вогнегасники, достоїнством яких є висока ефективність гасіння пожежі, збереження електричного устаткування. Розташовують вогнегасники на видних місцях, на висоті не більше як 1,5 м від полу. Виробничі приміщення мають запасні виходи. Двері повинні мати освітлений надпис «Запасний вихід». План евакуації вивішується на видному місці у основного виходу із приміщення.

3.4 Профілактика професійних захворювань

Оскільки робота з блокчейн-технологіями часто вимагає тривалого перебування в сидячому положенні, важливо забезпечити наступні заходи для профілактики професійних захворювань:

- Регулярні перерви: Впровадження регулярних перерв сприяє зниженню фізичної та візуальної втоми. Це дозволяє працівникам відпочивати та регенерувати, підтримуючи високу продуктивність протягом робочого дня.
- Оптимізація робочого простору: Налаштування робочого простору для мінімізації стресових і перенапружених рухів. Це включає ергономічну організацію робочого місця, таку як правильне розташування монітора, столу та стільця.
- Навчання правильній поставі: Проведення тренінгів та семінарів для навчання працівників правильній поставі і технікам ефективного чергування видів діяльності, щоб зменшити ризик розвитку професійних захворювань.

Впровадження комплексних заходів щодо безпеки при роботі з блокчейн-технологіями є ключовим для охорони праці. Це включає не тільки технічні засоби захисту, а й організаційні заходи. Важливо забезпечити, щоб кожен

працівник був добре обізнаний з потенційними ризиками, міг ефективно використовувати методи мінімізації цих ризиків, та мав доступ до необхідних засобів для забезпечення своєї особистої безпеки та здоров'я.

ВИСНОВКИ

У ході виконання дипломної роботи, метою якої було дослідження застосування технології блокчейн у різних галузях, було розглянуто основні аспекти та переваги технології, а також виклики, які вона ставить перед сучасними системами безпеки, фінансами та управлінням даними. Основні результати роботи:

1 Визначено ключові характеристики та принципи роботи блокчейну, такі як децентралізація, прозорість та незмінність даних.

2 Аналізовано застосування блокчейну в різних сферах, включаючи фінанси, охорону здоров'я та логістику.

3 Досліджено технології шифрування, що використовуються у блокчейні, такі як хешування та асиметричне шифрування.

4 Розглянуто алгоритми консенсусу, які є основою для валідації транзакцій у блокчейні, включаючи Proof of Work і Proof of Stake.

5 Виявлено потенційні виклики і обмеження блокчейну, такі як масштабування та використання ресурсів.

6 Оцінено можливості впровадження смарт-контрактів у різних додатках та їхній вплив на автоматизацію та зменшення потреби у посередниках.

7 Проведено аналіз стандартів токенів та їх застосування у створенні цифрових активів та економік, зокрема, ERC-20, ERC-721 та ERC-1155.

8 Розглянуто новітні розробки у сфері L2 рішень, що дозволяють знизити вартість транзакцій та підвищити масштабованість мережі.

9 Аналізовано імплементацію блокчейну в управлінні цифровими правами та автентифікації, підкреслюючи його значення у захисті інтелектуальної власності та особистих даних.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

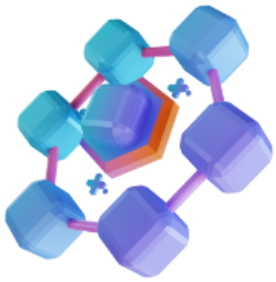
1. Березовський І. В., Шевченко О. О. Основи програмування на Python: навчальний посібник. - Київ: КНУ, 2020. - 324 с.
2. Іванченко О. М. Основи криптографії: навчальний посібник. - Київ: НАУ, 2018. - 288 с.
3. Кулініч В. П. Введення в блокчейн: навчальний посібник. - Харків: ХНУ, 2020. - 312 с.
4. Міщенко Т. В. Використання блокчейн у фінансових технологіях: монографія. - Київ: КНЕУ, 2019. - 299 с.
5. Павлюк М. М., Ткаченко В. Г. Основи криптовалют: навчальний посібник. - Львів: ЛП, 2021. - 275 с.
6. Пилипчук В. В. Програмування на Solidity: підручник. - Харків: ХАІ, 2020. - 360 с.
7. Тарасов Д. О. Розробка смарт-контрактів: підручник. - Київ: НТУУ "КПІ", 2020. - 333 с.
8. Харченко В. В. Безпека блокчейн: монографія. - Харків: ХНУРЕ, 2021. - 377 с.
9. Шимон О. В., Глушко І. В. Взаємодія з блокчейн-мережами: навчальний посібник. - Львів: ЛНУ, 2021. - 322 с.
10. Яковенко С. І. Криптографія для блокчейн: підручник. - Київ: НАУ, 2019. - 290 с.
11. Buterin V. A Next-Generation Smart Contract and Decentralized Application Platform. - Ethereum White Paper, 2014. - 36 p. - [Електронний ресурс] – Режим доступу до ресурсу: https://ethereum.org/en/whitepaper/
12. Greenspan G. MultiChain Private Blockchain — White Paper. - 2015. - 36 p. - [Електронний ресурс] – Режим доступу до ресурсу: https://www.multichain.com/download/MultiChain-White-Paper.pdf
13. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. - [Електронний ресурс] – Режим доступу до ресурсу: http://bitcoin.org/bitcoin.pdf

14. Tapscott D., Tapscott A. Blockchain Revolution: How the Technology Behind Bitcoin and Other Cryptocurrencies is Changing the World. - Penguin, 2016. - 368 p. - [Електронний ресурс] – Режим доступу до ресурсу: https://www.blockchain-revolution.com
15. Wood G. Ethereum: A Secure Decentralized Generalized Transaction Ledger. - Ethereum Project Yellow Paper, 2014. - 33 p. - [Електронний ресурс] – Режим доступу до ресурсу: https://ethereum.github.io/yellowpaper/paper.pdf

ДОДАТОК А. Слайди мультимедійної презентації

РОЗРОБКА СКРИПТА ВЗАЄМОДІЇ З БЛОКЧЕЙН-МЕРЕЖАМИ

ДИПЛОМНИЙ ПРОЕКТ

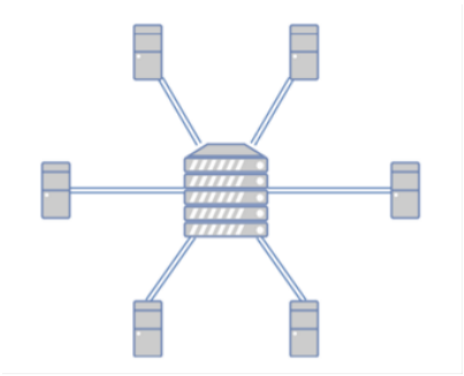


Керівник:
к.ф.н., доцент каф. КБ та ТЗІ ДУІТЗ Стайкуца С.В.

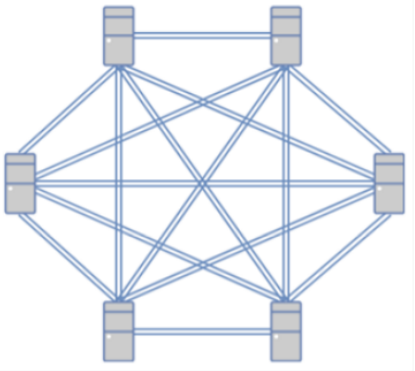
Виконав:
студент групи 4КБ-01 Кустуров А.С.

2024

Приклад з'єднання мереж



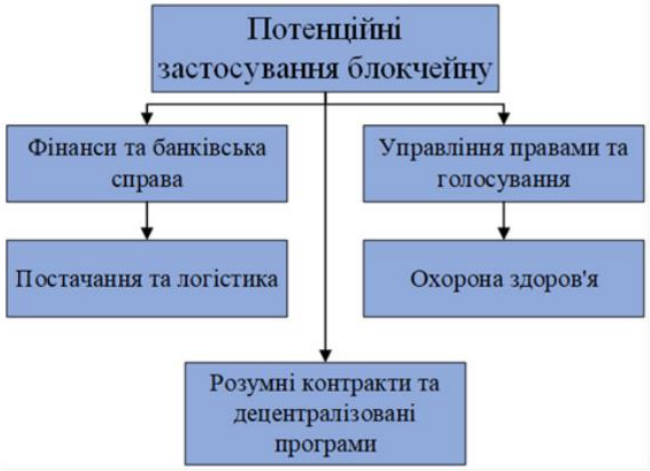
Традиційна централізована мережа



Розподілена мережа через технологію блокчейн

2

Приклад потенційного використання блокчейна

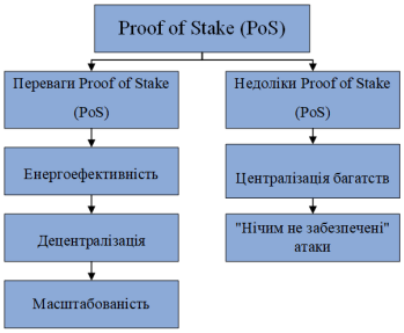


Потенційні застосування блокчейну.

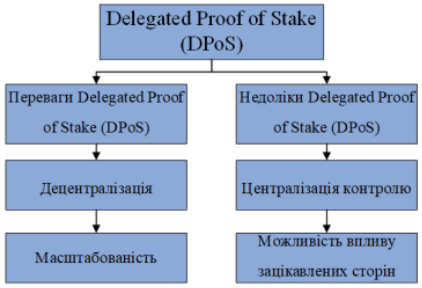
3

Технології консенсусу

Переваги та недоліки
Proof of Work (PoW).



Переваги та недоліки
Proof of Stake (PoS).



Переваги та недоліки
Delegated Proof of Stake (DPoS)

4

Мережі типу L1(Layer 1)

Назва	Тип консенсусу	Приблизна максимальна пропускна здатність (TPS)	Час створення блоку	Комісія за транзакцію
Bitcoin	PoW	7 транзакцій/сек	10 хвилин	$20 - $90
Ethereum	PoS	20-45 транзакцій/сек	12 секунд	$2 - $110
Solana	PoS	5000-7000 транзакцій/сек	0.4 секунди	$0.01 - $0.05
Binance Smart Chain (BSC)	PoSA	200 транзакцій/сек	3 секунди	$0.01 - $1

Порівняння характеристик різних блокчейн-платформ

5

Проблеми мереж типу L1(Layer 1)

Трилема блокчейна, ілюстрація взаємозалежності між безпекою, масштабуванням та децентралізацією в блокчейн-технологіях.

6

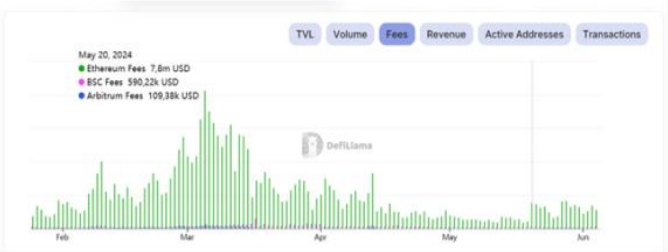
Мережі типу L2(Layer 2)

Мережа	Тип Rollup	Приблизна пропускна здатність (TPS)	Приблизний час створення блоку	Комісія за транзакцію
Arbitrum	Optimistic rollup	4,000 транзакцій/сек	10 секунд	$0.1-$0.5
Optimism	Optimistic rollup	2,000 транзакцій/сек	10 секунд	$0.1-$0.5
zkSync	ZK rollup	10,000 транзакцій/сек	2 секунди	$0.01-$0.05
StarkNet	ZK rollup	8,000 транзакцій/сек	10 секунд	$0.001-$0.005
Scroll	ZK rollup	10,000 транзакцій/сек	2 секунди	$0.01-$0.05

Порівняльний аналіз характеристик мереж L2 з різними типами rollups

7

Порівняння трьох платформ блокчейна за кількістю транзакцій та вартістю цих транзакцій

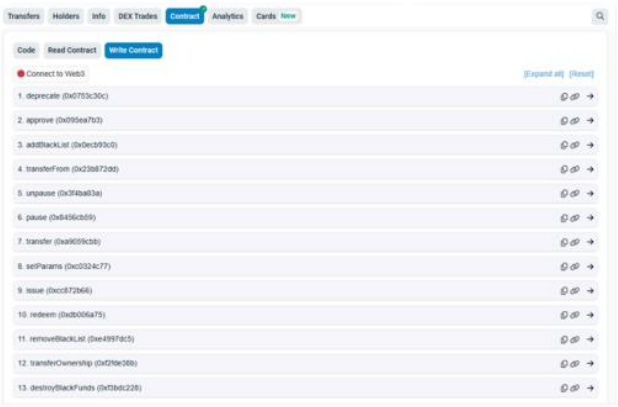


Порівняння плати за транзакції у трьох різних блокчейнах

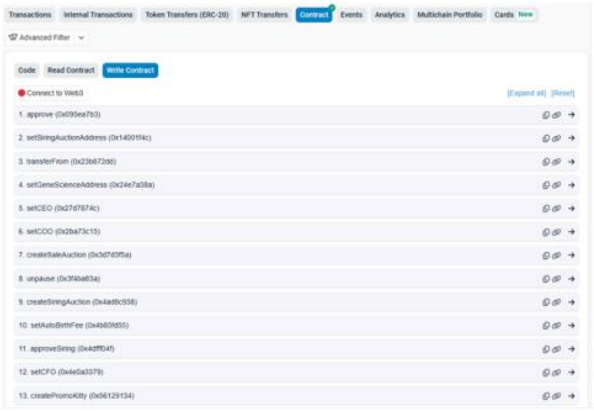
Кількість транзакцій у трьох різних блокчейнах

8

Класифікація стандартів токенів



Функції контракту ERC-20 токена, зосереджені на адміністративних операціях та переказах



Функції контракту ERC-721, зосереджені на управлінні активами та аукціонами

9

Характеристики та методи користування смартконтрактами

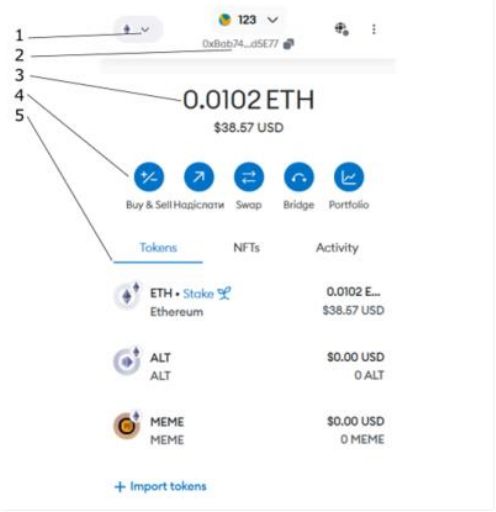
Основні характеристики смарт-контрактів

```
pragma solidity ^0.8.0;
contract SimpleStorage {
  uint storedData;
  function set(uint x) public {
    storedData = x;
  }
  function get() public view returns (uint) {
    return storedData;
  }
}
```

приклад коду мовою програмування Solidity

10

Криптовалютні гаманці



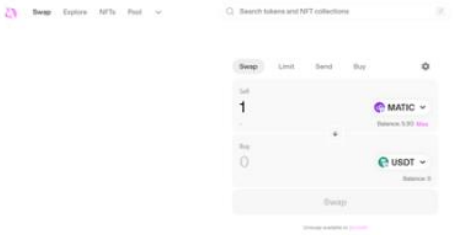
Інтерфейс некастодіального криптовалютного гаманця METAMASK.

Біржові гаманці

Апаратні гаманці

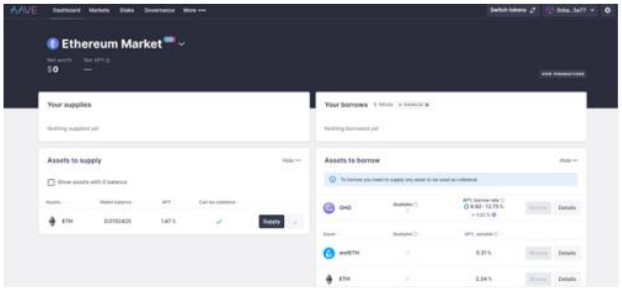
11

Види децентралізованих додатків



Інтерфейс користувача ринку ETHEREUM на платформі
AAVE для позичання та кредитування

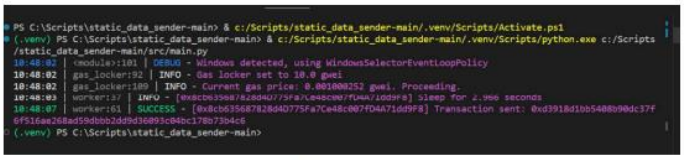
Інтерфейс для обміну криптовалютами на платформі UNISWAP.



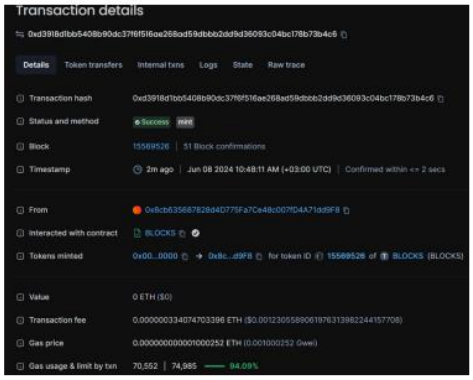
12

Схема скрипту та його результат роботи

Схема процесу відправки транзакцій за допомогою скрипта для автоматизації.



Логи скрипта



Інформація про виконану транзакцію

13

ДЯКУЮ ЗА УВАГУ!

ДОДАТОК Б. Програмний код мовою Python для розробленого скрипта

На зображенні представлена структура файлів та папок усередині директорії під назвою src. Вона включає папку core і кілька файлів. У папці core містяться такі файли Python (*.py): __init__.py, data_sender.py, utils.py та config.py, а також файли налаштувань config.yaml, main.py, текстовий файл private_keys.txt та файл залежностей requirements.txt.

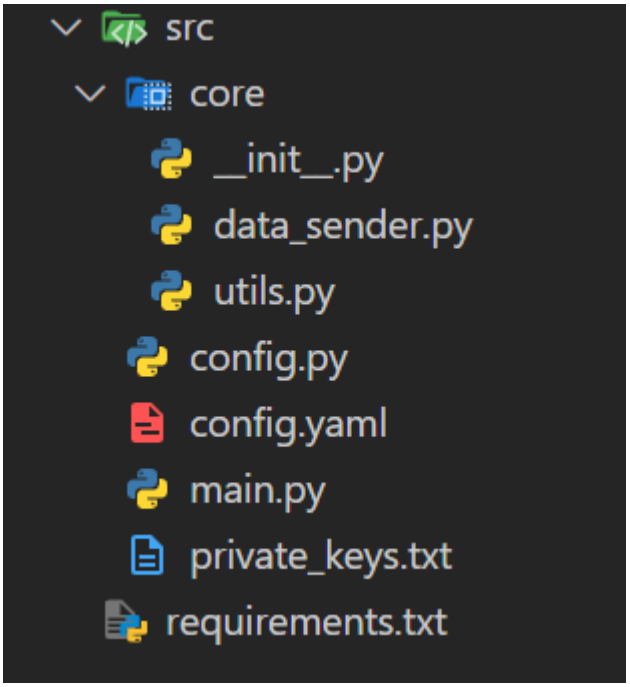


Рисунок 1. структура файлів та папок

1. **Файл __init__.py**

   Пустий

2. **Файл data_sender.py**

   import asyncio

   from eth_account import Account
   from hexbytes import HexBytes

```
from loguru import logger
from web3 import Web3
from web3.types import TxParams


class Web3Wrapper:
    def __init__(
        self,
        web3: Web3,
        private_key: str,
    ) -> None:
        self.web3 = web3
        self.__account: Account = Account.from_key(private_key)

        self.account_address = self.__account.address

    async def estimate_and_send_transaction(
        self,
        tx_params: TxParams,
        gas_multiplier: int | float = 1.05,
    ) -> HexBytes:

        if not self.web3.is_checksum_address(tx_params["to"]):
            tx_params["to"] = self.web3.to_checksum_address(tx_params["to"])

        if not tx_params.get("nonce"):
            tx_params["nonce"] = await self.web3.eth.get_transaction_count(
                self.account_address, "pending"
            )
```

```
            if not tx_params.get("from"):
                tx_params["from"] = self.account_address

            if not tx_params.get("chainId"):
                tx_params["chainId"] = await self.web3.eth.chain_id

            gas = await self.web3.eth.estimate_gas(tx_params)

            tx_params["gas"] = int(gas * gas_multiplier)
            signed_tx = self.__account.sign_transaction(tx_params)

            return                                                    await
self.web3.eth.send_raw_transaction(signed_tx.rawTransaction)

        @property
        async def eip1559_gas_price(self) -> dict[str, int]:

            base_fee = await self.web3.eth.gas_price

            max_priority_fee = await self.web3.eth.max_priority_fee

            return {
                "maxPriorityFeePerGas": max_priority_fee,
                "maxFeePerGas": (base_fee + max_priority_fee) * 2,
            }


    async def gas_locker(
        locker: asyncio.Lock,
        web3: Web3,
```

```
    gas_target: float | int
) -> None:

    # gas target from float to wei
    gas_target = gas_target * 1e9

    logger.info(f"Gas locker set to {gas_target / 1e9} gwei")

    await locker.acquire()

    while True:
        if (current_gas := await web3.eth.gas_price) > gas_target:
            if not locker.locked():
                await locker.acquire()

            logger.info(
                f"Current gas price: {current_gas / 1e9} gwei, "
                f"waiting for {gas_target / 1e9} gwei"
            )
            await asyncio.sleep(1)
        else:
            if locker.locked():
                locker.release()
                logger.info(f"Current gas price: {current_gas / 1e9} gwei. Proceeding.")
```

3. **Файл utils.py**

```
import sys
from loguru import logger
```

```
def configure_logger() -> None:
    logger.remove()

    logger.add(
        "logs/{time:MM_D}/{time:HH_mm}.log",
        format="{time:HH:mm:ss} | {function}:{line} | {level} - {message}",
    )
    logger.add(
        sys.stdout,
        format="<level>{time:HH:mm:ss}</level> | <lk>{function}</lk>:<lk>{line}</lk> | <level>{level}</level> - <magenta>{message}</magenta>",  # noqa: E501
        colorize=True,
    )
```

4. **Файл config.py**

```
import yaml
from pydantic import BaseModel


class Config(BaseModel):
    CONTRACT: str
    INPUT_DATA: str
    RPC: str
    WORKERS_COUNT: int
    SLEEP_BETWEEN_ACCOUNT_WORK: list[int]
    GAS_TARGET: float | int
```

```
    VALUE: int | float


def load_config() -> Config:
    with open("./src/config.yaml", "r") as f:
        config = yaml.safe_load(f)

    return Config.parse_obj(config)
```

5. **Файл config.yaml**

```
# Contract address
CONTRACT: ""

# Tx input data (example: 0xe9e05c42)
INPUT_DATA: ""

# tx value (float || int, not wei/gwei)
VALUE: 0

# Gas target (gwei in float || int)
GAS_TARGET: 6.2

# RPC
RPC: "https://rpc.ankr.com/eth"

# Range of sleep between account work (in seconds)
SLEEP_BETWEEN_ACCOUNT_WORK: [2, 7]

# Number of threads
```

WORKERS_COUNT: 2

6. **Файл main.py**

```
import asyncio
import sys
from loguru import logger

from web3 import Web3

from config import Config, load_config
from core.data_sender import Web3Wrapper, gas_locker
from core.utils import configure_logger
from web3.eth import AsyncEth
from random import uniform



async def worker(
    q: asyncio.Queue,
    web3: Web3,
    config: Config,
    locker: asyncio.Lock,
) -> None:
    while not q.empty():
        if locker.locked():
            await asyncio.sleep(1)
            continue

        private_key = await q.get()
```

```
        try:
            w3 = Web3Wrapper(
                web3=web3,
                private_key=private_key
            )
        except Exception as err:
            logger.error(f"Error while creating Web3 object: {err}") # noqa: E501
            continue

        sleep_time = round(uniform(*config.SLEEP_BETWEEN_ACCOUNT_WORK), 3)
        logger.info(f"[{w3.account_address}] Sleep for {sleep_time} seconds") # noqa: E501
        await asyncio.sleep(sleep_time)

        if (gas_price := await w3.web3.eth.gas_price) > config.GAS_TARGET * 1e9:
            logger.info(
                f"[{w3.account_address}] Gas price is too "
                f"high ({gas_price / 1e9} gwei), skipping"
            )
            q.put_nowait(private_key)
            continue

        try:
            tx_hash = await w3.estimate_and_send_transaction(
                tx_params={
                    "to": config.CONTRACT,
                    "data": config.INPUT_DATA,
                    "value": config.VALUE,
```

```
                **(await w3.eip1559_gas_price),
            }
        )
    except Exception as err:
        logger.error(f"[{w3.account_address}] Error while sending transaction: {err}") # noqa: E501
        continue

    logger.success(f"[{w3.account_address}] Transaction sent: {tx_hash.hex()}") # noqa: E501


async def main() -> None:
    config = load_config()

    web3 = Web3(
        Web3.AsyncHTTPProvider(config.RPC),
        middlewares=[],
        modules={"eth": (AsyncEth,)}
    )

    locker = asyncio.Lock()
    q = asyncio.Queue()

    asyncio.create_task(gas_locker(locker, web3, config.GAS_TARGET))

    with open("./src/private_keys.txt") as f:
        private_keys = f.read().splitlines()

    if not private_keys:
```

```
            logger.error("No private keys provided, exiting")
            sys.exit(1)

        [q.put_nowait(pk) for pk in private_keys]

        workers = [
            asyncio.create_task(worker(q, web3, config, locker))
            for _ in range(config.WORKERS_COUNT)
        ]

        await asyncio.gather(*workers)


    if __name__ == "__main__":
        configure_logger()

        # Windows fix for closed event loop
        if sys.platform.startswith("win"):

    asyncio.set_event_loop_policy(asyncio.WindowsSelectorEventLoopPolicy())
            logger.debug("Windows detected, using
    WindowsSelectorEventLoopPolicy")

        try:
            asyncio.run(main())
        except (KeyboardInterrupt, SystemExit):
            pass
```

7. **Файл private_keys.txt**

у цей файл потрібно вписати приватні ключі від гаманців

8. **Файл requirements.txt**

eth_account==0.9.0

hexbytes==0.2.2

loguru==0.7.0

pydantic==1.10.2

PyYAML==6.0.1

web3==6.7.0

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Кустурова Андрія Станіславовича

(прізвище, ім'я та по батькові)

Спеціальність: *123 "Комп'ютерна інженерія"*

Освітня програма: *«Безпека комп'ютерних систем і мереж»*

Тема дипломного проекту: Розробка скрипта взаємодії з блокчейн-мережами

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 88 сторінки. У пояснювальній записці розглянуто проблематика взаємодії зі блокчейн-мережами та запропоновано рішення у вигляді скрипта, що вирішує ланку питань. Графічна частина складається з слайдів мультимедійної презентації, які також містять креслення, передбачені технічним завданням. Якість виконання пояснювальної записки та графічної частини висока, розробка виконано в повному обсязі.*

б) самостійність роботи над проектом: *Протягом всього строку дипломного проектування та переддипломної практики здобувач освіти Кустуров А.С. виконував всі етапи розробки проекту, без порушення термінів. Роботу студент виконував в більшій мірі самостійно, з оглядом на рекомендації керівника та динамічно вносив потрібні коригування.*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Кустуров А.С. під час роботи над дипломним проектом проаналізував достатню кількість літературних джерел та матеріалів за даною тематикою.*

Вважаю, що теоретична підготовка дипломника якісна і він готовий до захисту дипломного проекту.

г) вміння розв'язувати виробничі та конструкторські питання _____

Під час дипломного проектування здобувач освіти Кустуров А.С. приймав самостійні рішення щодо вибору мереж та платформ, аналізував вимоги на різних етапах розробки, представляв проектні рішення, обґрунтовував вибір інструментів розробки, платформи та алгоритмів реалізації розробленого проекту.

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові керівника дипломного проекту _____

Стайкуца Сергій Володимирович

Місце роботи і посада керівника дипломного проекту _____

"Державний університет інтелектуальних технологій і зв'язку",
доцент кафедри кібербезпеки та технічного захисту інформації,
помічник декана факультету інформаційних технологій та кібербезпеки

Підпис _____

«____»_____2024 р.

ПІДПИС ПОСВІДЧУВ
НАЧАЛЬНИК ВІДДІЛУ
КАДРІВ ДУІТЗ
12.06.2024 р.

ВІДДІЛ КАДРІВ

ВСП «Одеський технічний фаховий коледж ОНАХТ»

РЕЦЕНЗІЯ

на дипломний проект (роботу) студента

Комп'ютерних систем відділення

Кустурова Андрія Станіславовича

(прізвище, ім'я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Керівник дипломного проекту (роботи)

Стайкуца Сергій Володимирович

(прізвище, ім'я та по батькові)

Тема дипломного проекту (роботи)

Розробка скрипта взаємодії з блокчейн-мережами

Обсяг розрахунково-пояснювальної записки 88 сторінок

Обсяг графічної частини проекту 14 аркушів

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) Висновок про ступінь відповідальності виконаного дипломного проекту (роботи) завданню

Робота відповідає технічному завданню до дипломного проекту. Виконана у відповідності з вимогами.

б) Характеристика виконання кожного розділу проекту ступеню використання дипломником останніх досягнень науки та техніки, передових методів на виробництві

При виконанні дипломного проекту студент продемонстрував уміння використовувати сучасні технології та розробки, уміння працювати з літературою.

в) Оцінка якості виконання графічної частини проекту (роботи) і пояснювальної записки

Графічна частина відповідає вимогам, виконана якісно та відображає основні елементи проектного рішення

г) Перелік позитивних якостей дипломного проекту (роботи)

Тема дипломного проекту є актуальною, виконана у достатньому обсязі, якісно, відповідно до поставленого завдання.

д) Основні недоліки дипломного проекту

У тексті пояснювальної записки відсутні посилання на використану літературу, слід було б відобразити більш детально результати поетапного виконання проекту у тексті пояснювальної записки. Однак, дані зауваження не применшуються актуальність роботи та якість її кінцевого виконання.

Оцінка розрахункової частини Відмінно

Оцінка графічної частини Відмінно

Загальна оцінка Відмінно

Прізвище, ім`я по батькові к.т.н. Селіванова Алла Віталіївна

Місце роботи і посада рецензента Одеський національний технологічний університет, декан факультету комп'ютерної інженерії, програмування та кіберзахисту

13.06.2024р

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Кустуров Андрій Станіславович,
здобувач освіти гр. 4КБ-01, та

Стайкуца Сергій Володимирович,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка скрипта взаємодії з блокчейн-мережами» (автор роботи – Кустуров А.С., керівник роботи – Стайкуца С.В.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2024 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Кустуров А.С. /

Керівник _____ / Стайкуца С.В. /

«10» червня 2024 р.

UNICHECK
by Turnitin

Ім'я користувача:
Катерина Григоріївна Краснокутська

ID перевірки:
1016335461

Дата перевірки:
08.06.2024 16:14:51 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
08.06.2024 16:17:31 EEST

ID користувача:
100011688

Назва документа: 4КБ-01 Кустуров Андрій

Кількість сторінок: 257 Кількість слів: 11093 Кількість символів: 76967 Розмір файлу: 2.45 MB ID файлу: 1016136107

12.3%
Схожість

Найбільша схожість: 10.2% з Інтернет-джерелом (https://card-file.ontu.edu.ua/server/api/core/bitstreams/4c0773a8-2d0.

12.3% Джерела з Інтернету | 99 ..... Сторінка 259

Не знайдено джерел з Бібліотеки

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел