Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

БАКАЛАВРСЬКА ДИПЛОМНА РОБОТА

на тему:

«Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології»

Виконав: студент 4 курсу, групи 1КІТС–20б
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем

_____ Стадник Р. Ю._____
(прізвище та ініціали)

Керівник: к.т.н., доц. кафедри МБІС
_____ Грицак А. В._____
(прізвище та ініціали)
«11» червня 2024 р.

Рецензент: к.т.н., доц. каф. ОТ
_____ Войцеховська О.В._____
(прізвище та ініціали)
«11» червня 2024 р.

Допущено до захисту
Голова секції УБ, кафедра МБІС
д.т.н., проф. Яремчук Ю. Є.
“11” червня 2024

Вінниця ВНТУ – 2024

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти І-й (бакалаврський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

_____ **д.т.н., проф. Яремчук Ю.Є.**
" 22 " березня 2024 р.

З А В Д А Н Н Я
на бакалаврську дипломну роботу студенту
Стаднику Ростиславу Юрійовичу
(прізвище, ім`я, по-батькові)

1. Тема роботи Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології
Керівник роботи Грицак Анатолій Васильович к.т.н., доцент кафедри МБІС
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "11" березня 2024року № 80
2. Термін подання студентом роботи 6 червня 2023р
3. Вихідні дані до роботи Електронні джерела, наукові статті, підручники та офіційні документи по темі роботи. Існуюче програмне забезпечення, яке відноситься до теми бакалаврської дипломної роботи.
4. Зміст текстової частини:
Для досягнення успішності виконання роботи потрібно було: дослідити предметну область обраної теми; здійснити аналіз технології, на якій базується розробка програмного забезпечення; здійснити вибір методів програмування; здійснити розробку програмного забезпечення; здійснити розробку програмного додатку.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)
Схема взаємодії користувача з ресурсом; схема роботи алгоритму програми; алгоритм роботи технології блокчейн; схема бази даних; UML-діаграма класів.
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ І	Грицак А. В. к.т.н., доцент кафедри МБІС		
Розділ ІІ	Грицак А. В. к.т.н., доцент кафедри МБІС		
Розділ ІІІ	Грицак А. В. к.т.н., доцент кафедри МБІС		

7.Дата видачі завдання 22 березня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	23.03.2024	02.04.2024	
2	Розробка захищеного веб-ресурсу	03.04.2024	12.04.2024	
3	Проектування веб-ресурсу	13.04.2024	25.04.2024	
4	Написання БДР на основі визначеної теми	26.04.2024	10.05.2024	
5	Тестування розробленого веб-ресурсу	11.05.2024	23.05.2024	
6	Попередній захист БДР	24.05.2024	02.06.2024	
7	Виправлення роботи	03.06.2024	11.06.2024	
8	Захист БДР	12.06.2024	17.06.2024	

Студент (підпис) Стадник Р.О. (прізвище та ініціали)

Керівник роботи (підпис) Грицак А.В. (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота присвячена розробці безпечної, масштабованої та зручної для користувача програмної системи на основі технології блокчейн для забезпечення ефективних та прозорих транзакцій з криптовалютами. Робота містить 63 сторінки, 24 ілюстрацію, 4 додатки та 46 джерел інформації.

Об'єктом розробки є система на основі блокчейну для здійснення транзакцій з криптовалютами. Метою роботи є створення децентралізованої мережі блокчейну, що інтегрує передові консенсусні механізми та рішення для масштабованості, забезпечуючи високу пропускну здатність, низьку затримку та енергоефективність без шкоди для прозорості, незмінності та безпеки.

У роботі застосовано методи криптографії, розподілених систем, консенсусних алгоритмів, а також сучасні веб-технології для розробки зручного для користувача інтерфейсу.

Результати впровадження демонструють високу продуктивність, відповідність нормативним вимогам, інтероперабельність та масштабованість системи. Новизна полягає у створенні комплексного рішення для децентралізованих фінансів на основі передових технологій блокчейну.

КЛЮЧОВІ СЛОВА: БЛОКЧЕЙН, КРИПТОВАЛЮТА, ДЕЦЕНТРАЛІЗОВАНА СИСТЕМА, КОНСЕНСУСНИЙ АЛГОРИТМ, МАСШТАБОВАНІСТЬ, БЕЗПЕКА, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, НОРМАТИВНА ВІДПОВІДНІСТЬ, ІНТЕГРАЦІЯ, РОЗПОДІЛЕНІ СИСТ

ABSTRACT

The thesis is devoted to the development of a secure, scalable, and user-friendly software system based on blockchain technology to ensure efficient and transparent transactions with cryptocurrencies. The work contains 63 pages, 24 illustrations, 4 appendix and 46 sources of information.

The object of development is a blockchain-based system for conducting transactions with cryptocurrencies. The purpose of the work is to create a decentralized blockchain network that integrates advanced consensus mechanisms and scalability solutions, providing high throughput, low latency, and energy efficiency without compromising transparency, immutability, and security.

The work uses cryptography, distributed systems, consensus algorithms, and modern web technologies to develop a user-friendly interface.

The implementation results demonstrate high performance, compliance with regulatory requirements, interoperability and scalability of the system. The novelty lies in the creation of a comprehensive solution for decentralized finance based on advanced blockchain technologies.

KEYWORDS: BLOCKCHAIN, CRYPTOCURRENCY, DECENTRALIZED SYSTEM, CONSENSUS ALGORITHM, SCALABILITY, SECURITY, USER INTERFACE, REGULATORY COMPLIANCE, INTEGRATION, DISTRIBUTED SYSTEMS.

ЗМІСТ

ВСТУП

Спосіб обробки даних значно змінився завдяки використанню технології блокчейн; він пропонує децентралізований, безпечний і прозорий підхід, який ставить під загрозу традиційні централізовані системи.

Крипто валюти швидко набували популярності у фінансовій сфері завдяки можливостям блокчейну, які полегшують однорангові транзакції без посередників. Це дослідження демонструє практичну значущість і актуальність технології блокчейн, що включає розробку комплексного веб-додатку для системи крипто валют на основі блокчейну.

Основна мета полягає в тому, щоб створити зручну для користувача платформу, яка дозволить проводити безперешкодні транзакції та підвищить довіру та прозорість у фінансових операціях. Ця робота використовує переваги блокчейну, такі як незмінність, децентралізація та криптографічна безпека, що дозволяє людям більше контролювати свої фінансові активи, одночасно зменшуючи ризики, пов’язані з централізованими системами. Задача полягає в розробці та впровадженні надійної та масштабованої програмної системи, яка полегшує створення, управління та валідацію мережі блокчейн, щоб забезпечити безпечні та ефективні транзакції криптовалюти.

Об’єктом дослідження є система криптовалюти на основі блокчейну, яка включає в себе інтерфейс користувача, структуру даних, архітектуру та алгоритми.

Дослідження стосується теоретичних основ технологій блокчейн, криптографії, процесів консенсусу та практичної реалізації децентралізованої фінансової екосистеми з акцентом на безпеку, надійність і користувацький досвід.

Цей проект є дуже важливим у наш час, коли довіра до традиційних фінансових установ зменшилася, а люди хочуть більшої автономії та прозорості у своїх фінансових операціях. Цей проект має на меті докорінно змінити те, як ми сприймаємо та здійснюємо фінансові операції, використовуючи передові можливості технології блокчейн, прокладаючи шлях до більш справедливого, безпечного та децентралізованого фінансового майбутнього.

РОЗДІЛ 1. АНАЛІЗ ЗАХИЩЕНИХ КОРПОРАТИВНИХ WEB-РЕСУРСІВ ДЛЯ НАДАННЯ ПОСЛУГ З ОНЛАЙН ТОРГІВЛІ НА ОСНОВІ БЛОКЧЕЙН ТЕХНОЛОГІЇ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Актуальність дослідження обраної технології

Предметом дослідження є технологія блокчейн і те, як вона використовується для створення децентралізованих фінансових систем, особливо криптовалютних мереж. Блокчейн — революційна технологія, яка вперше набула популярності завдяки появі біткоїна — з тих пір перетворився на багатогранну парадигму, яка вплинула на багато сфер [1].

На відміну від існуючих корпоративних і державних баз даних, де дані зберігаються в одному місці (централізовано), блокчейн складається з цифрової мережі розподілених по всьому світу комп'ютерів, так званих вузлів, які записують дані та історію транзакцій. Тож, очевидно, що блокчейн складається з блоків, які містять свій внутрішній функціонал (рис. 1.1) [2].

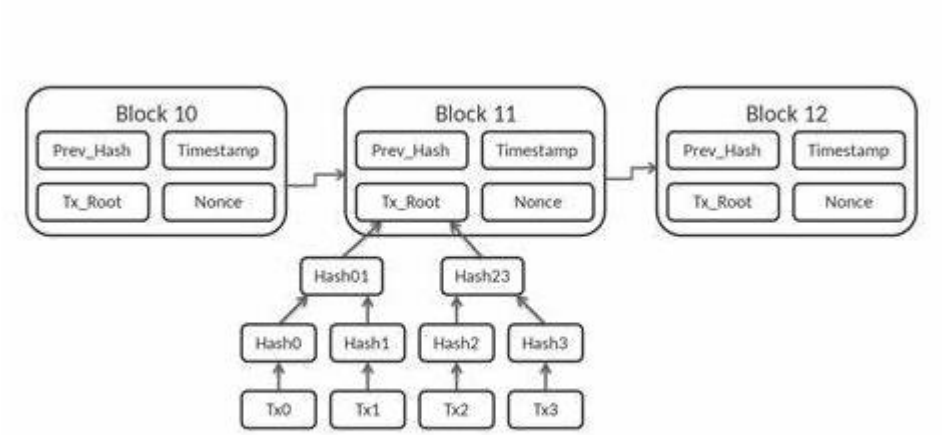


Рисунок 1.1 – Схема блоку з блокчейну

Вузли також відповідають за перевірку та зберігання всіх транзакцій з моменту створення бази даних. Оскільки система децентралізована, більшість вузлів повинні досягти спільної згоди. Якщо вузол хоче зробити власне оновлення,

інші вузли повинні проголосувати за це рішення, щоб переконатися, що оновлення є легітимним і безпечним. Після цього система оновлює записи до найсвіжіших і узгоджених оновлень на всіх вузлах одночасно. Цей процес голосування на вузлі відомий як «консенсус» [3].

Зважаючи на саму ідею функціоналу даної технології, навіть не беручи до уваги практичне застосування, можна сказати, що вона має перспективне майбутнє для застосування у більшості сфер, що критично впливають на життя людини.

Каталізатором, що поширення цієї технології, стала інноваційна концепція «Інтернету цінності» [4]. Цей термін означає використання Інтернету як засобу передачі та зберігання цінностей усіх видів, не тільки валют і цінних паперів, але й інтелектуальної власності, музики, мистецтва, результатів досліджень тощо. У цьому контексті блокчейн відіграє важливу роль, оскільки гарантує для цієї мети кілька важливих переваг, серед яких:

1. Ефективність – обмін цінностями відбувається в режимі, близькому до реального часу.

2. Децентралізація – це, мабуть, найважливіша концепція блокчейну, оскільки не існує третьої сторони або посередника, який би гарантував достовірність транзакцій та інформації, що вноситься до розподіленої бази даних. Насправді, для перевірки транзакцій і блоків використовується механізм консенсусу, який поділяється і приймається всіма вузлами мережі. З цієї причини блокчейн визначають як систему, що не потребує довіри.

3. Розподілена структура – оскільки, дані реплікуються на всіх вузлах мережі і географічно розподілені по всьому світу, система має високу стійкість до збоїв або кібератак. Оскільки кожен вузол має копію всієї бази даних, немає єдиної точки відмови, і будь-яке пошкодження однорангового вузла в мережі не впливає на роботу всієї системи, на відміну від централізованих систем, де кібератака або відключення дата-центру може перервати роботу всієї системи. Таке резервування забезпечує високу доступність і надійність всієї мережі.

4. Прозорість – дані, що містяться в блокчейні, доступні всім вузлам мережі, а у випадку загальнодоступних мереж вони також вільно доступні будь-кому, хто отримує доступ до мережі, не беручи при цьому активної участі в підтримці життєдіяльності системи. Ця особливість надає блокчейну властивість бути транс-батьківським, і тому довіра до системи гарантується цією особливістю.

5. Незмінність – після того, як дані були записані в блокчейн, їх вкрай складно змінити, оскільки це вимагатиме величезних витрат обчислювальної потужності. Якщо, наприклад, ви хочете змінити транзакцію, яка була записана 10 блоків тому, хакеру доведеться змінити і перерахувати всі 10 блоків або перерізати ланцюжок і зробити недійсними наступні 9 блоків. Насправді це майже неможливо реалізувати на багатьох блокчейнах.

6. Високозахищена структура за задумом – всі транзакції в блокчейні зашифровані, що забезпечує високу цілісність мережі.

7. Програмованість – у блоки можна включати інструкції, які запускають специфічні дії при настанні певних умов.

8. Незаперечність – рішення про достовірність інформації приймається не лише вузлом, а через захищений механізм збору консенсусу в мережі, що робить неможливим оскаржити вибір, зроблений більшістю.

Після перелічених позитивних властивостей, можна зробити висновок, що дана технологія є гіппер-передовою, адже виконує нівелювання більшості проблем, які впливають на безпеку даних, їх цілісність та можливу втрату.

В основі технології блокчейн лежать кілька основних ідей. Забезпечення цілісності та безпеки даних, що зберігаються в блокчейні, залежить від криптографії. Криптографічні хеш-функції, такі як SHA-256 (рис. 1.2) [5], створюють унікальні цифрові відбитки для кожного блоку, що створює нерозривний ланцюжок даних. Крім того, асиметричні алгоритми шифрування, такі як алгоритм цифрового підпису еліптичної кривої (ECDSA). Принцип роботи ECDSA полягає в тому, що аналізується еліптична крива і вибирається точка на кривій. Ця точка множиться на інше число, створюючи таким чином нову точку на кривій. Нову точку на кривій дуже важко знайти, навіть маючи у своєму

розпорядженні початкову точку. Складність ECDSA означає, що ECDSA є більш захищеним від сучасних методів злому шифрів. Крім того, що ECDSA є більш захищеним від сучасних методів атак, він також пропонує ряд інших переваг [6].

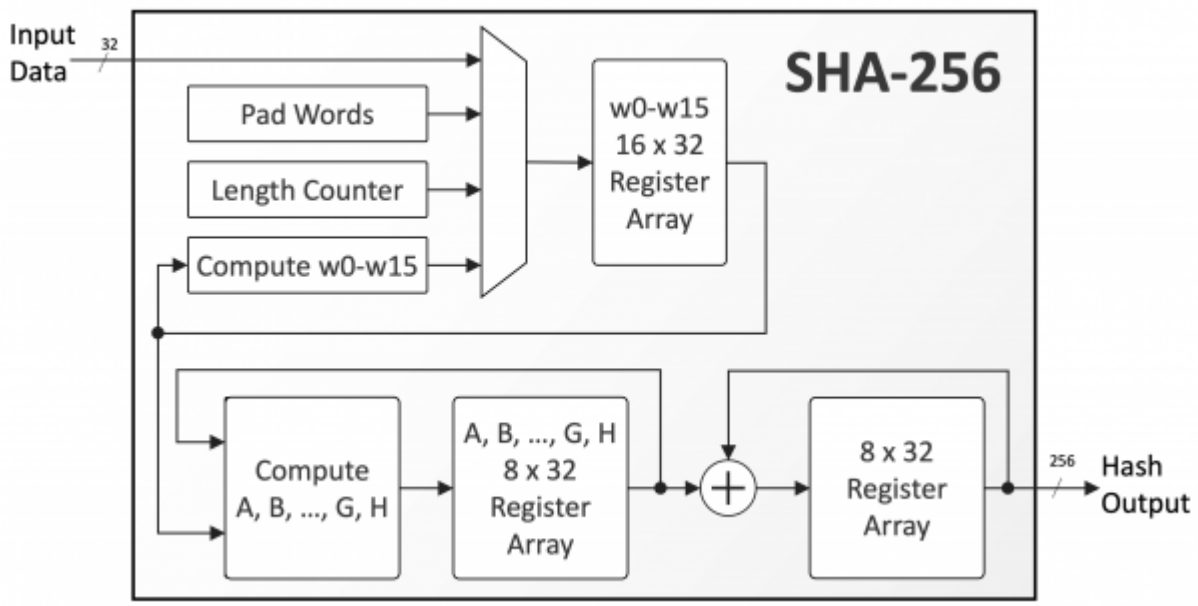


Рисунок 1.2 – Схема роботи SHA-256

Механізм консенсусу є ключовою концепцією технології блокчейн. У мережі немає центрального органу управління, тому алгоритми консенсусу використовуються для досягнення згоди між розподіленими вузлами щодо дійсності транзакцій і стану реєстру. Proof-of-Work (PoW), який використовується в біткоїні, і Proof-of-Stake (PoS), який має на меті зменшити витрату енергії, є двома поширеними механізмами консенсусу.

Криптовалюта – це цифрова валюта, призначена для роботи в якості засобу обміну через комп'ютерну мережу, яка не залежить від будь-якого центрального органу, такого як уряд або банк, для її підтримки або обслуговування[7].

Індивідуальні записи про власників монет зберігаються в цифровій книзі, яка є комп'ютеризованою базою даних, що використовує стійку криптографію для захисту записів про транзакції, контролю за створенням додаткових монет і перевірки передачі права власності на монети [8].

Незважаючи на те, що цей термін використовується для опису багатьох створених взаємозамінних токенів блокчейну, криптовалюти не вважаються валютами в традиційному розумінні, і в різних юрисдикціях до них застосовуються різні правові режими, в тому числі їх відносять до товарів, цінних паперів і валют.

На практиці криптовалюти зазвичай розглядаються як окремий клас активів. Деякі криптосистеми використовують валідатори для підтримки криптовалюти. У моделі підтвердження частки власники надають свої токени в якості застави. В обмін на це вони отримують повноваження над токеном пропорційно до суми, яку вони внесли. Як правило, ці стейкхолдери з часом отримують додаткові права власності на токени через мережеві збори, новостворені токени або інші подібні механізми винагороди[9].

Блокчейн, поки що, є єдиною новою технологією, яка суттєво вплинула на операції та послуги фінансового сектору. Технологія блокчейн є новою революцією в банківській та фінансовій сферах, і її слід широко впроваджувати. Хоча технологія блокчейн спочатку була розроблена для полегшення створення криптовалют, її поточні потенційні застосування виходять далеко за рамки цього. Технологія розподіленого реєстру, відома як блокчейн, прокладає шлях до безпечного та безпосередницького переказу цифрових валют, активів та інформації [10].

Законодавство та міркування, пов’язані з криптовалютою та децентралізованими фінансами (DeFi), все ще розробляються, і існує потреба в більшому розумінні та прийнятті серед широкої громадськості та політиків з різних держав світу.

Дослідження в цій галузі зосереджувалися на алгоритмах консенсусу, рішеннях для масштабування та розробці смарт-контрактів. Дослідники також вивчали вплив криптовалют на економіку, їхній вплив на традиційні фінансові установи та розробку децентралізованих програм (DApps) на основі мереж блокчейн [11].

Отже, не зважаючи на значний прогрес, дослідження та розробки в таких сферах, як користувацький досвід, безпека та відповідність нормативним вимогам, все ще потрібні. Розробка зручної, безпечної та сумісної криптовалютної системи

на основі блокчейну є метою цієї роботи, щоб долучитися до цих зусиль. Крім того, її мета полягає в тому, щоб подолати диспропорцію між технологічними досягненнями та реальним використанням цих технологій у фінансовому секторі.

1.2. Аналіз літератури обраної технології

Багато важливих тем і напрямків досліджень було виявлено під час аналізу існуючої літератури щодо розробки криптовалютних систем на основі блокчейну. Це включає основну технологію, економічні та регуляторні наслідки, а також проблеми практичної реалізації.

Дослідники та розробники вивчали фундаментальну технологію, яка лежить в основі блокчейну та криптовалют. Основні роботи, такі як «Біла книга» Сатоші Накамото про біткоїн, заклали основу для розуміння основних принципів блокчейну, таких як криптографічне хешування, розподілені реєстри та механізми консенсусу, такі як «доказ роботи» (PoW) та «доказ частки» (PoS). Згодом дослідження заглибилися в тонкощі цих ідей і запропонували покращення та альтернативні методи.

Наприклад, численні дослідження були зосереджені на збільшенні масштабованості існуючих мереж блокчейн, які часто не можуть обробляти великі транзакції. Запропоновані рішення включають позамережеві транзакції, шардінг і рішення для масштабування другого рівня, такі як Lightning Network. Платіжний протокол «рівня 2» Lightning Network (LN) базується на блокчейні криптовалют, таких як біткойн. Він був запропонований як рішення проблеми масштабування біткоїна і призначений для забезпечення швидких транзакцій між вузлами-учасниками, тобто незалежними членами мережі. Це система, яка працює на одному рівні і дозволяє здійснювати мікроплатежі криптовалюти через мережу платіжних каналів, які мають два напрямки, не надаючи нікому повноважень щодо зберігання коштів [12].

Проте, варто зазначити, що багато з цих рішень все ще в процесі розробки, і, звичайно, для їх впровадження необхідні додаткові дослідження, щоб визначити, наскільки ефективними вони є та як вони впливають на безпеку.

У поточному дослідженні також розглядаються методи досягнення консенсусу, які є більш енергоефективними. Алгоритм Proof-of-Work у біткоїні критикують за те, що він потребує багато енергії. Це спонукає дослідників вивчати альтернативні алгоритми, такі як гібридні, Proof-of-Stake та Proof-of-Authority. Незважаючи на те, що ці альтернативи є багатообіцяючими щодо енергоефективності, питання безпеки та децентралізації ще не вирішено.

Економічні та регулюючі наслідки криптовалют також досліджуються.

Блокчейн можна розглядати як механізм інституційного управління, що створює розподілені реєстри інформації та керує ними. Він слугує структурою управління, що полегшує безпечний потік довіреної інформації вздовж ланцюгів постачання для споживачів, фірм та урядів. Наприклад, у ланцюгах поставок ця технологія дозволяє відстежувати активи в режимі реального часу, від видобутку сировини до доставки кінцевому споживачеві, полегшуючи точну перевірку викидів вуглецю [13].

Також можна розглядати як вплив на сектор аеропортів. По-перше, аеропортова галузь відіграє важливу роль у SCM. Дійсно, аеропорти є критично важливими вузлами в системі повітряного транспорту, а отже, і в сполученні територій. Вони також є частиною авіаційного сектору. По-друге, аеропортове господарство зазнало важливих змін завдяки технології блокчейн завдяки впровадженню нових і технологічно просунутих додатків. Нарешті, аеропортовий сектор має все більш визнаний екологічний, соціальний та економічний вплив, що вимагає значної уваги в пошуку технічних і управлінських рішень для досягнення стійкої роботи, одночасно долаючи негативні наслідки технологічних збоїв. Тож, застосування даної технології до цієї сфери покращить безпекову складову [14].

Державні органи повинні створити достатню правову базу та правила для транзакцій криптовалюти та систем, заснованих на блокчейні. У той час як деякі юрисдикції вважають криптовалюту більш лояльною, інші запровадили жорсткі правила або прямо заборонили її використання. Через відсутність єдиної глобальної стандартної бази широке використання технології блокчейн у фінансовому секторі створює значні проблеми.

Дослідники вивчали різні аспекти створення зручних і безпечних криптовалютних гаманців, бірж і децентралізованих додатків (dApps). Користувацький досвід і адаптація були визнані ключовими компонентами успіху будь-якої системи, заснованої на блокчейні. Інтуїтивно зрозумілі інтерфейси, надійні засоби захисту та ефективна обробка транзакцій є критично важливими, щоб залучити основних користувачів [15].

Інтеграція технології блокчейн із застарілими системами та сучасною фінансовою інфраструктурою - ще одна тема, яка привертає увагу. Науковці дослідили перешкоди та потенційні засоби для забезпечення сумісності між мережею блокчейн та традиційними методами оплати [16].

До прикладу, у традиційному платіжному потоці від трьох до п’яти сторін здійснюють одну транзакцію. Вони створюють «стек платежів». Щоб створити довіру, ці різні сторони повинні співпрацювати. Вони контролюють переказ коштів і перевіряють можливість проведення транзакцій. Тим не менш, довіра має свою ціну, яку врешті-решт сплачують торговці. Від транзакції кожна сторона в стеку платежів отримує невелику суму.

Типова транзакція включає в себе перевірку платіжним процесором банку-емітента, чи можна списати кошти з картки клієнта. Після підтвердження транзакції, яке відбувається протягом декількох мілісекунд, торговець має гарантію того, що він отримає оплату пізніше. Протягом наступних днів кошти переказуються від банку-емітента до банку-еквайра.

Традиційний стек передбачає численні комісії. Карткові мережі та інші сторони також можуть підвищувати свої комісії. Нещодавно, у вересні 2019 року, Visa додала фіксовану комісію в розмірі 0,02 євро для торговців, які використовують 3D-Secure, що все частіше вимагається згідно з новим законодавством PSD2.

Продавці, які використовують традиційний стек, стикаються ще з однією проблемою. Крім того, швидкість транзакцій також може бути проблемою. Незважаючи на те, що валідація відбувається за мілісекунди, продавцю можуть знадобитися дні, перш ніж гроші потраплять до банку. Це не підходить для малого

та середнього бізнесу, який значною мірою залежить від грошових потоків, щоб розраховуватися з роботодавцями та постачальниками.

Для продавців ситуація ще гірша, якщо вийти за рамки карткових платежів. У Сполучених Штатах платіжний цикл між підприємствами та підприємцями займає в середньому 34 дні, при цьому 47% рахунків оплачуються запізненням.

Потенціал блокчейну очевидний, коли ми розглядаємо стек платежів як інструмент для побудови довіри, оскільки він може повністю усунути стек. Споживачі переказують гроші безпосередньо ритейлерам, а децентралізована мережа перевіряє транзакцію.

Блокчейн обіцяє значно знизити витрати і підвищити швидкість для ритейлерів. Якщо на гаманці недостатньо коштів, мережа відхиляє транзакцію; посередники не потрібні, щоб перевірити, чи можна доставити кошти чи ні. Кошти з'являються за кілька хвилин після підтвердження транзакції. Єдиною витратою є мережева комісія, яка покривається безпосередньо клієнтом [17].

Вони також взяли до уваги дотримання законодавства та безпеку даних.

Увага до захисту конфіденційності блокчейн-реєстрів і пов'язаних з ними регуляторних технологій зросла в міру того, як розподілені додатки, засновані на технології блокчейн, стають все більш поширеними в повсякденному житті. Регулювання приватних транзакцій – це те, що поточні основні рішення здебільшого ігнорують на користь надійного шифрування адрес і вмісту транзакцій у блокчейні. Крім того, обмежена кількість систем, що піддаються моніторингу, має такі проблеми, як надмірна централізація і вузький фокус на регуляторному змісті. У цій роботі представлено метод збереження конфіденційності блокчейну, який використовує шифрування на основі атрибутів (ABE) і докази з нульовим знанням (zk-SNARK) для багаторівневого регулювання з метою подолання цих недоліків.

По-перше, наша технологія забезпечує збереження конфіденційності блокчейну всередині моделі облікового запису за допомогою zk-SNARK, що дозволяє приховати адреси та дані транзакцій користувачів. По-друге, шифрування на основі атрибутів використовується для забезпечення багаторівневої нормативної бази, яка підтримує заходи захисту конфіденційності, дозволяючи при цьому

вибірково розкривати вміст транзакцій. Нарешті, ми досліджуємо безпеку запропонованої схеми і порівнюємо її з альтернативними схемами, розповідаючи про її переваги щодо конфіденційності, безпеки та регуляторних можливостей. Ми також пропонуємо початкову експериментальну оцінку ефективності схеми. Підсумовуючи, можна сказати, що план демонструє надійну конфіденційність, покладаючись на математичні демонстрації за допомогою zk-SNARK для забезпечення захисту при повному збереженні матеріалу. Крім того, він забезпечує децентралізовану регуляторну владу, широке регуляторне покриття і багаторівневе регулювання, побудоване на основі захисту приватності [18].

Відсутність єдиної міжнародної бази правил для криптовалют і систем, заснованих на блокчейні, створює труднощі для широкого впровадження та міжнародних транзакцій. Адже, криптовалюти стали лідерами у розвитку технології блокчейн, але блокчейн - це набагато більше, ніж просто платформа для цифрових активів. Блокчейн полегшує переміщення будь-яких цінностей, інформації або даних. Як наслідок, вони можуть використовувати цифрові документи, що зберігаються в захищеному від несанкціонованого доступу реєстрі, для заміни паперових. Як державні, так і комерційні організації досліджують потенційні переваги, які технологія блокчейн може принести їхнім відповідним службам. Результатом цього є «приватизація» даної технології [19].

Ця приватно розроблена, але широко доступна технологія почала використовуватися комерційним приватним сектором. Це призвело до появи приватних дозволених блокчейнів. Блоки належать до одного суб’єкта або консорціуму компаній, який має право обмежувати доступ до них, змінювати їх і використовувати протоколи з обмеженим доступом. Крім того, існують гібридні державно-приватні блокчейни, у яких вузли з приватним доступом можуть бачити весь блокчейн, тоді як інші вузли не можуть, або навпаки.

Важливим аспектом розвитку блокчейну також є користувацький досвід, адже багато людей, що мали змогу працювати з даною технологією навіть не розуміють, як вона працює, більше того, багато інтерфейсів не пристосовані для простого користувача.

Окрім потенційних переваг або фінансових транзакцій, технологія розробки блокчейну вже зараз впливає на реальний світ. Вона є технологічною основою для низки інновацій, що стали можливими завдяки публічним блокчейнам, таким як Ethereum та Bitcoin, які висвітлюють такі рішення, як децентралізовані автономні організації (DAO), не взаємозамінні токени (NFT) та децентралізовані фінанси (DeFi).

Соціальні мережі на основі технології блокчейн пропонують децентралізовану альтернативу для користувачів, щоб обмінюватися контентом і взаємодіяти безпечно, приватно і без інвазивного спостереження, яке є результатом централізованої влади компанії або організації.

Стейблкоіни - криптоактиви, забезпечені фіатними резервами та фінансовими активами, а також заставою, - розроблені для прив'язки до фіатних валют, таких як долар США або євро. Вони зменшують складність і високі витрати, пов'язані з грошовими переказами в традиційній фінансовій системі.

Розробка онлайн-карт, які децентралізовані, керовані громадою та оновлюються в режимі реального часу, є одним із варіантів використання технології блокчейн.

Використання технології блокчейн у системах та програмах заохочення споживачів є одним з найвідоміших прикладів її застосування.

Завдяки токенам або NFT, які надають клієнтам корисні функції, такі як доступ до товарів, послуг, подій або ексклюзивні привілеї за лояльність до бренду чи організації, технологія блокчейн дозволяє бізнесу розробляти більш ефективні та прозорі програми лояльності та винагород.

З появою децентралізованих фінансів, або DeFi, клієнти тепер можуть довіряти коду, а не людині, коли йдеться про фінансові послуги, завдяки технології блокчейн і смарт-контрактам.

1.3. Аналіз та порівняння існуючих web-ресурсів для онлайн торгівлі

Bitcoin Core - це проект з відкритим вихідним кодом, який підтримує і випускає клієнтське програмне забезпечення Bitcoin під назвою «Bitcoin Core» (рис.

1.1). Це прямий нащадок оригінального програмного клієнта Bitcoin, випущеного Сатоші Накамото після того, як він опублікував знамениту біткоїн-документацію. Bitcoin Core складається як з «повновузлового» програмного забезпечення для повної перевірки блокчейну, так і з біткоїн-гаманця. Проект також підтримує супутнє програмне забезпечення, таке як бібліотека криптографії libsecp256k1 та інші, розміщені на GitHub [20].

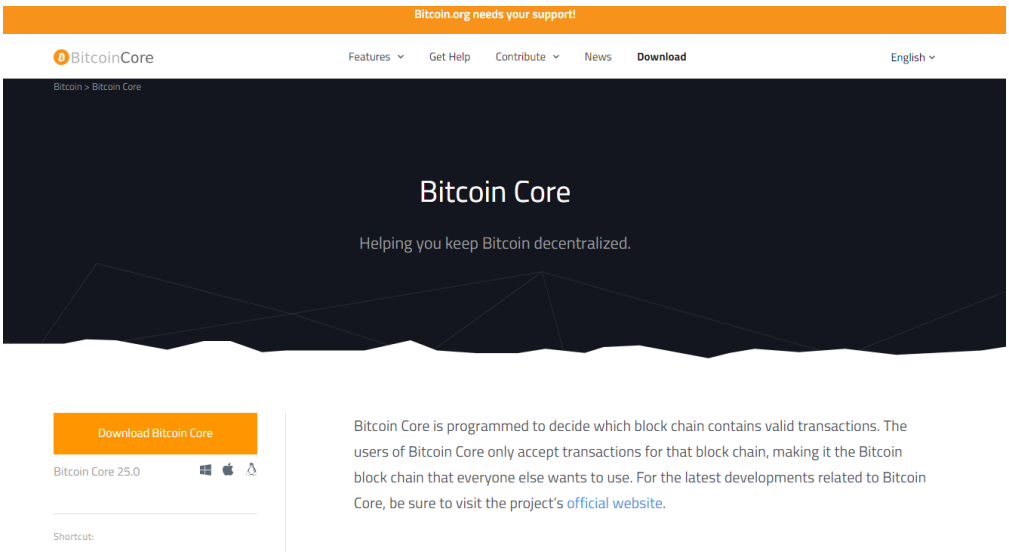


Рисунок 1.1 – Головна сторінка сайту команди Bitcoin Core

Механізми та характеристики Bitcoin Core:

1. При першому запуску Bitcoin Core завантажує блоки з інших вузлів, починаючи з найпершого (генезис) блоку. Цей процес може зайняти значний час, оскільки обсяг даних великий.

2. Вузол перевіряє кожен блок і транзакцію, щоб переконатися в їхній коректності.

3. Нові транзакції, створені користувачами, передаються до мережі через вузли Bitcoin Core.

4. Вузол перевіряє транзакції на правильність і передає їх іншим вузлам.

5. Нові блоки, створені майнерами, надходять до вузлів Bitcoin Core, які перевіряють їх на відповідність правилам мережі.

6. Якщо блок коректний, він додається до блокчейну, і вузол починає працювати з наступним блоком.

7. Bitcoin Core підтримує мережу, перевіряючи транзакції та блоки, що допомагає забезпечити її стабільність і безпеку.

Ethereum - це мережа комп'ютерів по всьому світу, які дотримуються набору правил, що називається протоколом Ethereum (рис. 1.2). Мережа Ethereum є основою для спільнот, додатків, організацій та цифрових активів, які кожен може створювати та використовувати.

Користувач можете створити обліковий запис Ethereum з будь-якого місця і в будь-який час і досліджувати світ додатків або створити свій власний. Основна інновація полягає в тому, що користувач може робити все це, не довіряючи центральному органу влади, який може змінити правила або обмежити його доступ [21].

Спочатку Ethereum використовував конкурентний процес перевірки доказів роботи, подібний до того, що використовується в біткоїні. Після кількох років розробки, у 2022 році Ethereum остаточно перейшов на систему підтвердження частки (proof-of-stake), яка використовує набагато менше обчислювальних потужностей та енергії [22].

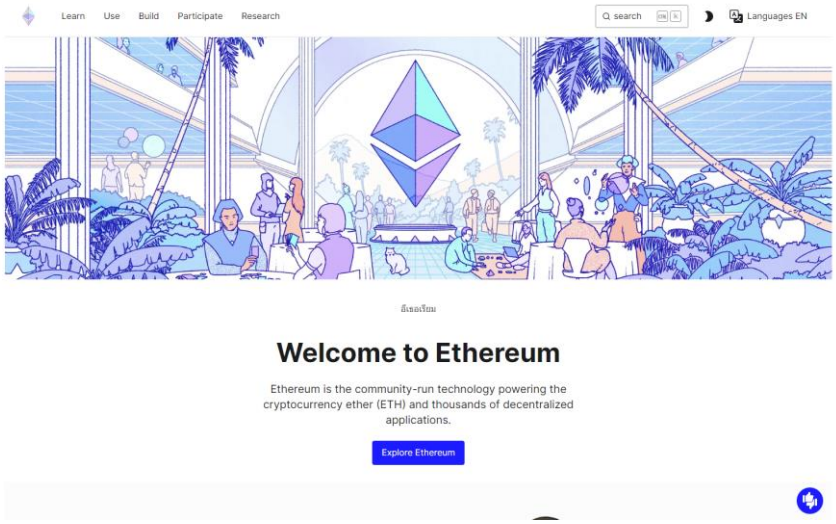


Рисунок 1.2 – Головна сторінка сайту команди Ethereum

Proof-of-stake відрізняється від proof-of-work тим, що не вимагає енергоємних обчислень, так званого майнінгу, для підтвердження блоків. Він використовує протокол фіналізації під назвою Casper-FFG і алгоритм LMD Ghost, об'єднані в механізм консенсусу під назвою Gasper. Gasper відстежує консенсус і визначає, як валідатори отримують винагороду за роботу або караються за нечесність або відсутність активності [23].

Механізми та характеристики Ethereum:

1. Користувачі ініціюють транзакції, які можуть включати передачу ETH, виклики смарт-контрактів або створення нових смарт-контрактів.

2. Транзакції надсилаються до мережі та розповсюджуються між вузлами.

3. Раніше Ethereum використовував Proof of Work (PoW), але перейшов на Proof of Stake (PoS) з оновленням Ethereum 2.0.

4. У PoS валідатори обираються для створення нових блоків і підтвердження транзакцій, базуючись на кількості ETH, які вони ставлять у заставу (stake).

5. Смарт-контракти виконуються автоматично, коли їхні умови виконуються. EVM виконує контракти на всіх вузлах мережі, забезпечуючи однаковий результат для всіх.

6. Контракти можуть взаємодіяти з іншими контрактами та змінювати свій стан у відповідь на отримані транзакції.

7. Блокчейн зберігає стан кожного смарт-контракту та баланси користувачів. Вузли мережі оновлюють свої копії блокчейну з кожним новим блоком.

Ripple - це система валових розрахунків у режимі реального часу, мережа обміну валют і грошових переказів, відкрита для фінансових установ по всьому світу, створена американською технологічною компанією Ripple Labs Inc (рис. 1.3). Випущена в 2012 році, Ripple побудована на розподіленому протоколі з відкритим вихідним кодом і підтримує токени, що представляють фіатну валюту, криптовалюту, товари або інші одиниці вартості, такі як милі для часто літаючих пасажирів або хвилини мобільного зв'язку. Ripple має на меті забезпечити «безпечні, миттєві та майже безкоштовні глобальні фінансові транзакції будь-якого

розміру без повернення платежів». У книзі використовується власна криптовалюта, відома як XRP [24].

Механізми та характеристики роботи Ripple:

1. Користувач ініціює транзакцію, наприклад, переказ коштів з одного банківського рахунку на інший через мережу Ripple.

2. Транзакція передається до мережі серверів Ripple, які перевіряють її коректність.

3. Сервери (вузли) у мережі Ripple виконують роль валідаторів, які перевіряють транзакції та оновлюють реєстр.

4. Валідатори використовують механізм консенсусу, щоб погодити новий стан реєстру. Це відбувається кожні кілька секунд, що забезпечує дуже швидке підтвердження транзакцій.

5. Ripple дозволяє проводити транзакції між різними валютами. Якщо необхідно конвертувати одну валюту в іншу, система використовує XRP як проміжний актив для забезпечення ліквідності.

6. Наприклад, якщо користувач хоче обміняти євро на долари США, Ripple може конвертувати євро в XRP, а потім XRP в долари США.

7. Транзакції в мережі Ripple займають всього кілька секунд, на відміну від традиційних банківських переказів, які можуть займати кілька днів.

8. Комісії за транзакції у мережі Ripple дуже низькі, що робить її привабливою для великих фінансових установ.

Stellar - це децентралізований публічний блокчейн, який надає розробникам інструменти для створення досвіду, який більше схожий на готівку, ніж на криптовалюту (рис. 1.4). Мережа швидша, дешевша та набагато енергоефективніша, ніж більшість систем на основі блокчейну. Екосистема Stellar розроблена таким чином, щоб мати довготривалий вплив на реальний світ [25].

Механізми та характеристики Stellar:

1. Користувачі ініціюють транзакції через анкори або безпосередньо в мережі, надсилаючи кредити чи XLM іншим користувачам.

2. Транзакції надходять до мережі вузлів Stellar, які перевіряють їх на коректність.

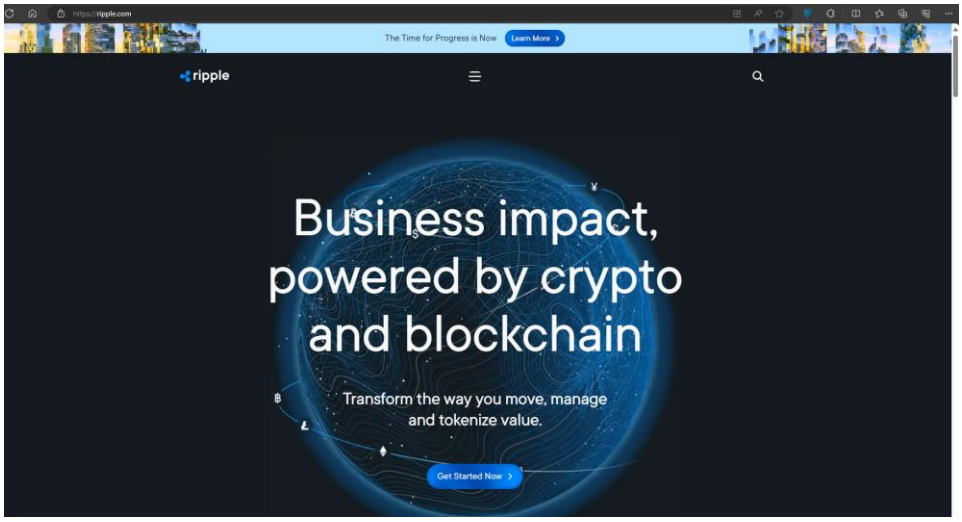


Рисунок 1.3 – Гостьова сторінка системи Ripple

3. Вузли мережі Stellar використовують SCP для досягнення консенсусу щодо валідності транзакцій. Цей процес займає кілька секунд.

4. Після досягнення консенсусу реєстр оновлюється, додаючи нові транзакції.

5. Stellar дозволяє обмінювати різні валюти через свій децентралізований обмінний механізм. Користувачі можуть створювати ордери на обмін валют у реєстрі.

6. Обмін відбувається автоматично, використовуючи найкращі доступні ордери.

7. Люмени (XLM) використовуються як проміжний актив для полегшення обміну між різними валютами, забезпечуючи ліквідність і знижуючи витрати на конвертацію.

Coinbase Global, Inc, під брендом Coinbase - американська публічна компанія, яка управляє платформою для обміну криптовалют. Coinbase є розподіленою компанією, всі співробітники працюють віддалено. Це найбільша криптовалютна біржа в США за обсягом торгів [26]. Компанія була заснована в 2012 році Брайаном Армстронгом і Фредом Ерсамом. У травні 2020 року Coinbase оголосила, що закриває свою штаб-квартиру в Сан-Франциско, штат Каліфорнія, і переходить на

віддалений режим роботи, що стало частиною хвилі закриття штаб-квартир кількох великих технологічних компаній в Сан-Франциско після пандемії COVID-19 [27].

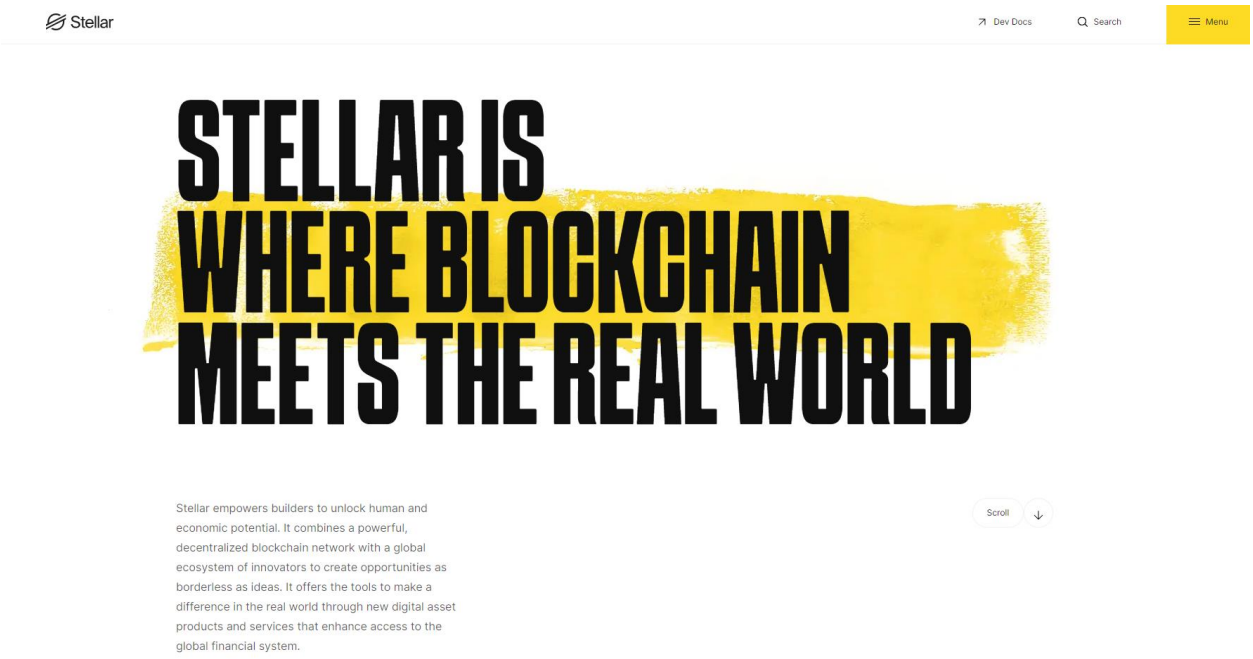


Рисунок 1.4 – Гостьова сторінка блокчейну Stellar

Кожен з розглянутих проектів має свої унікальні переваги та недоліки, що робить їх придатними для різних користувачів та сценаріїв використання. Bitcoin Core є найкращим вибором для збереження вартості, тоді як Ethereum пропонує величезні можливості для розробки децентралізованих додатків. Ripple підходить для швидких банківських платежів, тоді як Stellar акцентує увагу на фінансовій інклюзії. Coinbase, будучи централізованою біржею, пропонує зручність і широкий вибір криптовалют, але з меншою свободою через регуляторний контроль.

Дана робота присвячена вирішенню більшості проблем, що присутні в кожному вже існуючому рішенні. Тому важливою складової розробки чогось нового є аналіз попередніх результатів.

1.4. Висновки та постановка задач

У даному розділі було досліджено технологію Blockchain, її переваги та недоліки, специфіку використання. Також взято на спостереження використання

технологій на даний момент у світі, що вже передбачає лише розповсюдження технології.

Взято до уваги наявні конкретні приклади використання, такі як трейдингові платформи, а також засоби, що використовуються на державному рівні, а також проведено аналіз літератури та наявної інформації, що до фактичного використання блокчейну та проблем з подальшим розповсюдженням.

Беручи до уваги матеріали дослідження з впевненістю можна зробити висновок, що технологія буде зазнавати значних змін у покращенні та буде застосовуватись у все більшій кількості складових життя людини.

Для того, щоб забезпечити відповідність нормативним вимогам, заповнити прогалини та усунути недоліки в існуючих рішеннях, а також забезпечити ефективні та прозорі криптовалютні транзакції, необхідно розробити безпечну, масштабовану та зручну для користувача програмну систему, що використовує технологію блокчейн.

І звичайно для досягнення поставленої мети було визначено конкретні задачі роботи, що повпливають на отриманий результат:

1. Розробка структури та функціоналу ресурсу для онлайн торгівлі.
2. Розробка алгоритму функціонування web-системи для онлайн трейдингу.
3. Розробка захищеної бази даних web-ресурсу.
4. Вибір засобів програмування для розроблюваного додатку.
5. Розробка програмної частини захищеного web-додатку.
6. Тестування розроблено програмного засобу для онлайн трейдингу.

Хоча основною метою проекту є створення криптовалютної системи, важливо пам'ятати, що технологія блокчейн та ідеї, що лежать в її основі, мають більш широке застосування за межами фінансової індустрії. Тому отримана інформація та створені рішення можуть сприяти просуванню технології блокчейн у різних галузях та сферах застосування.

Розглядаючи визначену постановку проблеми та досягаючи окреслених цілей, проект має на меті надати комплексне та практичне рішення, яке подолає

розрив між теоретичними основами технології блокчейн та її реальним впровадженням у фінансовому секторі.

Зважаючи на те, що постановка задачі була пропрацьована і окреслено конкретні цілі реалізації проекту, можна стверджувати, що з допомогою теоретичних змінних у впровадженні блокчейну та прикладами впровадження, а також ретельному процесу розробки, дотриманням нормативних вимог, а також достатньому акценту на користувацькому досвіді отримана система дозволить приватним особам зручно контролювати та продуктивно маніпулювати своїми фіанансами.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ЗАХИЩЕНОГО WEB-РЕСУРСУ З НАДАННЯ ПОСЛУГ В ОНЛАЙН ТОРГІВЛІ НА ОСНОВІ БЛОКЧЕЙН ТЕХНОЛОГІЇ

Даний розділ передбачає опис розробки проекту захищеного корпоративного web-ресурсу з надання послуг з онлайн торгівлі, впровадження алгоритму роботи програми, алгоритму функціональності технології блокчейну. Також, передбачає розробку бази даних, що забезпечить якісне та безпечне збереження даних.

2.1 Розробка структури та функціоналу ресурсу для онлайн торгівлі

Беручи до уваги, що метою розробки є створення захищеного web-ресурсу для надання послуг з онлайн торгівлі з використанням технології блокчейн, то, звичайно, головним завданням роботи є створення такого додатку, що надасть користувачам можливість проводити трейдингові операції, слідкувати за просуванням та змінами різних валют та безпечно зберігати свої активи.

Для створення повноцінного web-додатку потрібно дотриматись створення необхідних елементів його структури:

1. Наявність можливості створення облікового запису для користувача.
2. Можливість авторизації.
3. Можливість спостереження за поточним станом криптовалюти.
4. Змога переглядати історію транзакцій.
5. Здатність до покупки криптовалюти.
6. Можливість передавання криптовалюти іншим користувачам, лише існуючим.

Отже, проаналізувавши всі пункти можна визначити, що для створення потрібного додатку потрібні такі елементи візуального контенту:

1. Гостьова сторінка.
2. Вікно реєстрації.
3. Вікно авторизації.
4. Панель навігації по додатку.

5. Вкладка з інформацією про поточний стан криптовалюти та історією транзакцій.

6. Вкладка з допуском до покупки визначеної крпитовалюти.

7. Вкладка, що передбачає функцію передачі, продажу криптовалюти іншим користувачам платформи.

Вище описаний функціонал було відображено на схематичному прикладі, який покроково описує послідовність дій (рис. 2.1).

Але звертаючи увагу на описану схему, можна стверджувати, що при запуску web-ресурсу, його гість потрапляє на початкову сторінку, яка передбачає ознайомлення з інтерфейсом, на ній користувач може почати роботу з ресурсом, тобто зареєструватись на трейдинговій платформі, або ж вже за наявності акаунту лише авторизуватись.

Тоді вже залежно від того яку дію користувач обирає відбувається певний процес, хоча, в кінцевому результаті користувач все одно отримує доступ до всіх можливостей ресурсу, хоча є і вийняток, бо у користувача можуть з’явитись проблеми з авторизацією, тоді доведеться або відновлювати акаунт або звертатись до системної підтримки.

Після успішних авторизаційних дій, дані користувача зберігаються у базі даних та після цього відкриваються можливості до таких вже вище описаних переваг:

1. Доступ до трейдингових операцій.

2. Доступ до спостереження за рухом криптовалюти.

3. Можливість перегляду вже завершених транзакцій.

Отже, проведений аналіз дав зрозуміти, що надана схема повноцінно описує взаємодію користувача з сервісом та при її перегляді всі аспекти і незрозумілі деталі можуть переглянуті та використані на практиці як інструкція.

Варто зауважити, що на схемі зображено розгалуження варіантів, відповіді програми, що можуть вплинути на подальший шлях користувача.

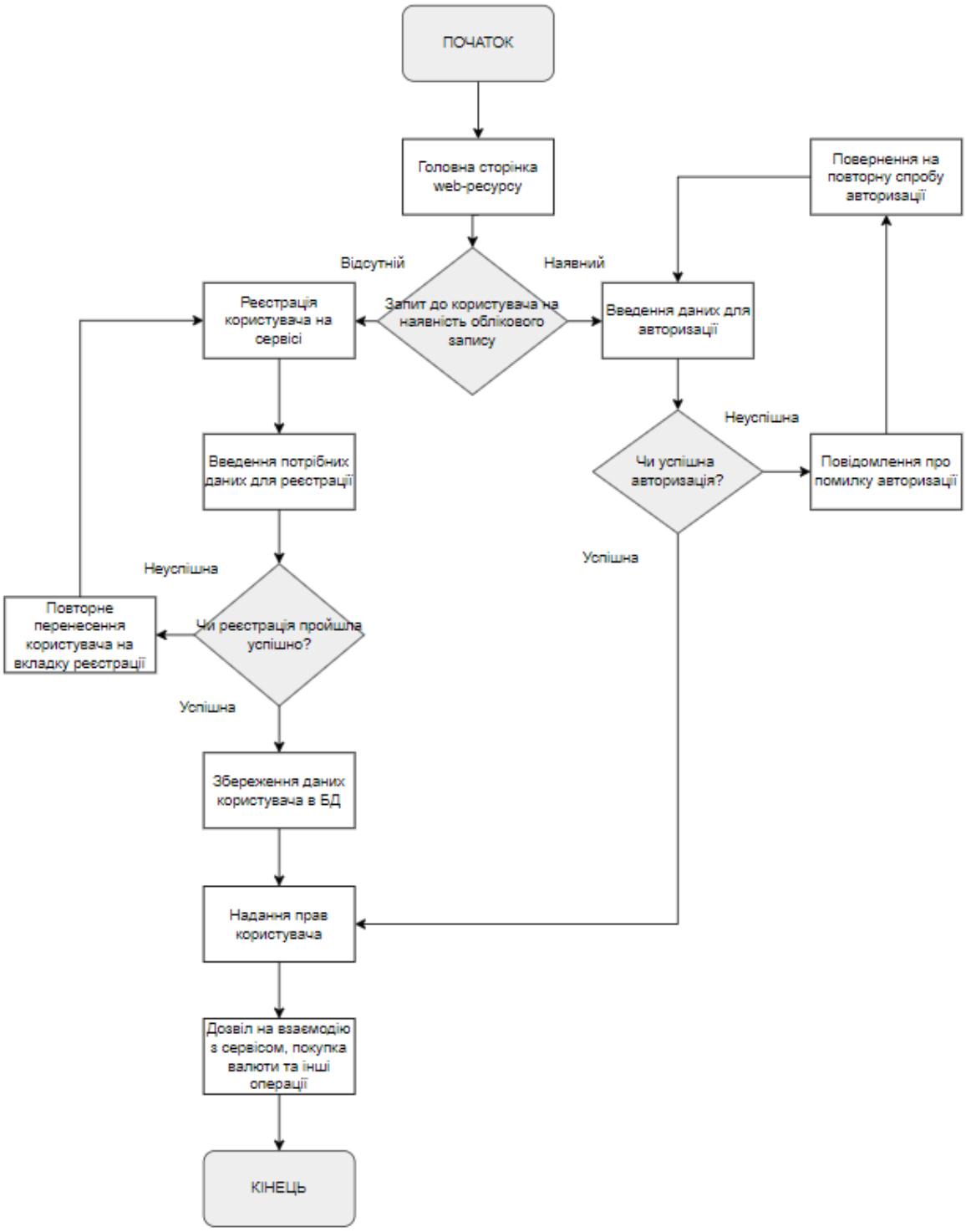


Рисунок 2.1 – Схема взаємодії користувача з ресурсом для онлайн торгівлі

Інші засоби, що допомагають підтримувати працездатність ресурсу, такі як процес блокчейну, внутрішні перевірки форм, механізми запису користувачів та взаємодії з базою даних, залишаються поза межами спостереження пересічного користувача.

2.2 Розробка алгоритму функціонування та впровадження блокчейн технології web-системи для онлайн трейдингу

Звичайно, для успішної реалізації задуманого web-ресуру для онлайн трейдингу потрібно розробити відповідний алгоритм роботи програми, для подальшого удосконалення та розуміння об’єму програми та конкретних завдань, які потрібно виконати для досягнення успіху у розробці.

Так як онлайн ресурс для трейдингу має забезпечувати доступ до певних функцій для користувачів, які подано у переліку:

1. Логінізація.

2. Реєстрація.

3. Надання можливості передавання та покупки валюти.

Ґрунтуючись на основних функціях ресурсу для онлайн трейдингу було розроблено основні етапи алгоритму (рис. 2.2):

Крок 1. Перший крок містить створення та перенесення на форму реєстрації (GET /register), тобто коли користувач відкриває сторінку реєстрації, сервер повертає HTML-форму для заповнення даних користувача (ім'я, email, ім'я користувача, пароль). Після цього цього відбувається введення даних (POST /register) та опісля заповнення форми реєстрації користувач надсилає свої дані на сервер для обробки. Наступним етапом першого кроку є перевірка валідності та існування, тобто сервер перевіряє, чи всі дані заповнені правильно і чи не існує вже користувача з таким ім'ям користувача в базі даних.

Крок 2. Наступний крок передбачає збереження в БД та логін користувача, тобто якщо дані валідні і користувач новий, сервер зберігає дані користувача в базі даних і автоматично входить користувача в систему. Після цього відбувається перенаправлення на Dashboard (GET /dashboard), це означатиме успішну реєстрацію, тобто після неї та входу користувача, його буде перенесено на сторінку Dashboard, де він може переглянути свій баланс та іншу інформацію. Також, звичайно алгоритм передбачає форма логіну (GET /login).

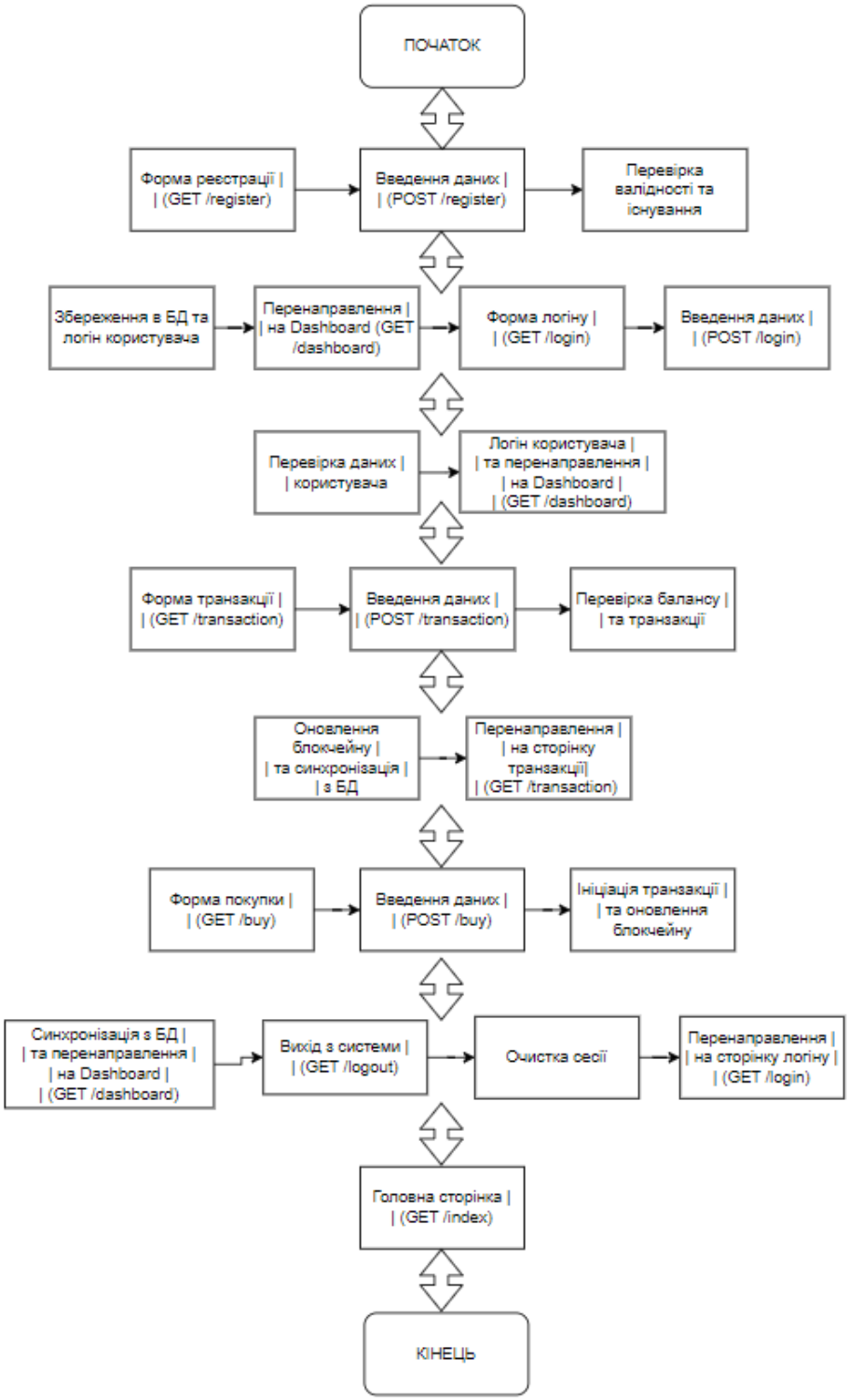


Рисунок 2.2 – Схема алгоритму роботи web-сервісу для онлайн трейдингу

Крок 3. Третій крок передбачає перевірку даних користувача, отже, сервер перевіряє, чи існує користувач з таким ім'ям користувача і чи пароль правильний. Після успішної логінізації користувача відбувається перенаправлення на Dashboard

(GET /dashboard). Якщо дані правильні, сервер входить користувача в систему та перенаправляє його на Dashboard.

Крок 4. Наступний крок передбачає створення форми транзакції (GET /transaction). Коли користувач відкриває сторінку транзакції, сервер повертає HTML-форму для заповнення даних транзакції (одержувач та сума). Далі відбувається введення даних (POST /transaction). Після заповнення форми транзакції користувач надсилає свої дані на сервер для обробки. Згодом проводиться перевірка балансу та транзакції. Сервер перевіряє, чи достатньо коштів у користувача для здійснення транзакції і чи є всі інші умови транзакції валідними.

Крок 5. П’ятий крок проводить оновлення блокчейну та синхронізує з БД. Якщо транзакція валідна, сервер оновлює блокчейн новим блоком і синхронізує ці дані з базою даних. Після завершення транзакції користувача перенаправляють на сторінку транзакції (GET /transaction), для перегляду оновленого балансу.

Крок 6. Наступним етапом є формування вікна покупки (GET /buy). Тобто можливість для користувача відкриття сторінки покупки і повернення HTML-форми з заповненими даними. Введення даних (POST /buy) - Після заповнення форми покупки користувач надсилає свої дані на сервер для обробки. Далі проводиться ініціація транзакції та оновлення блокчейну, коли сервер обробляє покупку як транзакцію, оновлює блокчейн новим блоком і синхронізує ці дані з базою даних.

Крок 7. Передостаннім кроком є синхронізація з БД та перенаправлення на Dashboard (GET /dashboard), що дозволяє користувачу переглянути фактичний баланс. Якщо користувач провів усі потрібні йому операції, то в нього є можлвість до виходу з системи (GET /logout). Після чого відбувається очистка сесії.

Крок 8. Останнім кроком подається пернаправлення користувача на початкову сторінку для поадльших функцій по web-додатку для онлайн торгівлі.

Аналізуючи розроблений алгоритм можна стверджувати, що розроблюваний захищений корпоративний web-ресурс для онлайн трейдингу має хороший фундамент в образі алгоритму, що значно полегшить практичну реалізацію.

Деталізація кожного кроку та розбиття його та менші пункти допоможе уникнути помилок при програмуванні та правильно передавати виконання шляхів для відповідності розробленій схемі.

Окреслений сервіс для онлайн трейдингу на основі блокчейн технології, розробці якого присвячена дана робота, функціонуватиме за допомогою модульної та масштабованої архітектур, а за їх наявності, звичайно, також буде проведено проектування конкретної технології блокчейн, що функціонуватиме в програмі та описаної у розробленому алгоритмі для web-ресурсу.

Важливо зауважити, що блокчейн який буде використовується у розробці є основою для запису і взагалі реалізації будь-якої криптовалютної системи. Цей компонент відповідає за перевірку та обробку транзакцій, підтримання узгодженості в мережі та забезпечення цілісності даних за допомогою надійних механізмів консенсусу.

Важливу роль у понятті блокчейну відіграє криптографія, що є основою даної технології і без її розгляду неможливо починати розробку алгоритму для даної технології. Так як криптографія в основному відноситься до секретів в буквальному сенсі. Тому технології криптографії спрямовані на забезпечення повної або псевдо-анонімності.

Основними цілями застосування криптографії є запобігання подвійним витратам, гарантування безпеки учасників і транзакцій, а також мінімізація впливу центральної влади на операції. Криптографія має кілька застосувань для різних видів інформації. Іноді вона допомагає забезпечити безпеку певних мережевих транзакцій. Однак вона також може використовуватися для підтвердження передачі цифрових активів і токенів. Для наочнішого прикладу того, як криптографія функціонує у технології блокчейн можна звернутись до вже інсуючої ілюстрації (рис. 2.3) [29].

Мережа блокчейн побудована на фундаментальних принципах криптографії та розподілених систем. Вона використовує передові криптографічні алгоритми, такі як SHA-256 для хешування та алгоритм цифрового підпису еліптичної кривої (ECDSA) для цифрових підписів, щоб захистити транзакції та встановити право

власності на криптовалютні гаманці. Також вже раніше розглянуті механізми консенсусу, що використовуються, засновані на алгоритмі Proof-of-Work (PoW), який передбачає розв'язання криптографічних головоломок, що вимагають великих обчислювальних витрат, для перевірки нових блоків і запобігання подвійним витратам [29].

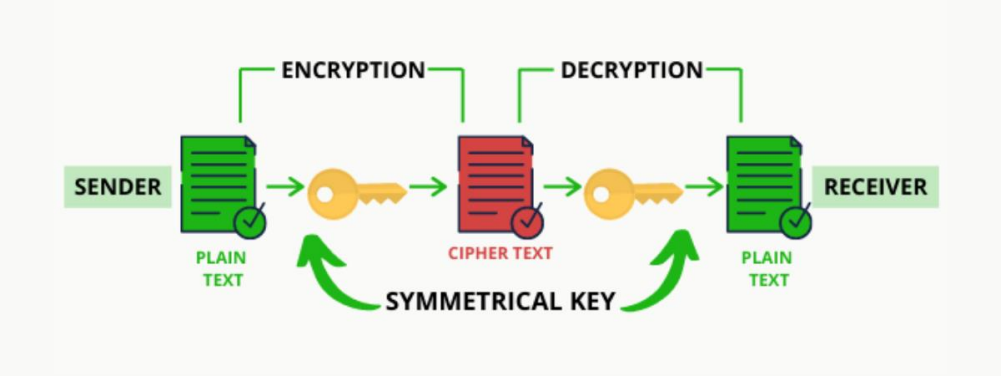


Рисунок 2.3 – Візуальне представлення криптографічного впливу на блокчейн

Для повноцінної та правильної реалізації задуманого рішення потрібно детально продумати розробку блокчейну системи.

Досягнення мети допоможуть реалізувати три основні компоненти:

1. Окремий клас Block, що представлятиме окремий блоку у блокчейні та буде містити конструктор, який буде приймати параметри попереднього хешу, даних та номер блоку. Також важливою складовою є хеш-метод, що повертатиме SHA256 хеш блоку. Та підсумковий метод str, що повертатиме вихідну інформацію.

2. Цілісний клас Blockchain, який представляє ланцюг блоків, тобто сам блокчейн. Містить атрибут класу, який визначає складність майнингу, конструктор, що буде ініціювати побудову ланцюга, метод додавання, видалення та оновний метод майнингу.

3. Базисна функція, що ініціалізуватиме блокчейн, міститиме список даних, які додаються до блоків, майнить блоки, друкує блоки, а в кінці перевіряє цілісність та валідність блокчейну.

Завдяки хитрощам мережі не потрібно створювати пари між кожним користувачем. Наприклад, якщо користувач А має канал з користувачем В, а

користувач С має канал з користувачем В, але користувач А не має каналу з користувачем А, гроші все одно можуть безперешкодно переміщуватися між усіма учасниками мережі. Процеси оплати користувачів, які застосовуються до адрес Lightning, дуже схожі на звичайні адреси Bitcoin [30].

Потрібно звернути увагу на основи модульного дизайну, що забезпечить масштабованість веб-додатку для онлайн трейдингу та забезпечить потрібний рівень безпеки, а також дасть властивість повторного використання створеного коду (рис. 2.4).

```
    # Get one value from the table based on a column's data
    def getone(self, search, value):
        data = {}
        cur = conn.cursor()
        result = cur.execute(f"SELECT * FROM {self.table} WHERE {search} = ?", (value,))
        data = result.fetchone() if result else None
        cur.close()
        return data

    # Delete a value from the table based on column's data
    def deleteone(self, search, value):
        cur = conn.cursor()
        cur.execute(f"DELETE FROM {self.table} WHERE {search} = ?", (value,))
        conn.commit()
        cur.close()

    # Delete all values from the table
    def deleteall(self):
        self.drop()
        self.__init__(self.table, *self.columnsList)

    # Remove table from SQLite
    def drop(self):
        cur = conn.cursor()
        cur.execute(f"DROP TABLE {self.table}")
        cur.close()
```

Рисунок 2.4 – Приклад модульного дизайну архітектури

Кожен компонент блокчейну має дотримуватися стандартних галузевих практик безпеки, таких як перевірка вхідних даних, санітарна обробка та шифрування, щоб зменшити ризик вразливостей і потенційних атак.

Зважаючи на розглянуті аспекти та конкретні компоненти коду, що будуть слугувати основою для реалізації блокчейну для розроблюваного захищеного корпоративного онлайн сервісу для трейдингу було сформовано схему роботи блокчейну (рис. 2.5):

Крок 1. Першим кроком є ініціалізація блокчейну, тобто підготовка до створення нового ланцюга блоків.

Крок 2. Наступним кроком слугує створення об’єкту Blockchain, що проводить ініціалізацію порожнього ланцюга блоків.

Рисунок 2.5 – Алгоритм роботи технології блокчейну розробленого корпоративного та захищеного web-ресурсу для онлайн трейдингу

Крок 3. Ініціалізація бази даних також є важливим кроком у алгоритмі, що формує список рядкових значень, які будуть додані у блоки ланцюга.

Крок 4. Один з найважливіших кроків є майнинг блоків. Для кожного рядкового значення створюється новий блок з урахуванням номера, даних та хешу попереднього блоку. Виконується майнінг блоку, тобто знаходження нонса, при якому геш блоку задовольняє умовам складності. Після успішного майнінгу блок додається в ланцюг.

Крок 5. Перевірка валідності блокчейну, тобто перевіряється, чи хеші всіх блоків коректні та задовольняють умовам складності, далі проводиться еревірка полягає в порівнянні хешу попереднього блоку з записаним у поточному блоці хешем та перевірці відповідності умовам складності.

Крок 6. Шостим кроком окреслюється зміна даних блоку, при проведенні певної операції, тобто внесення змін до одного з блоків у ланцюгу.

Крок 7. Надалі відбувається повторний майнінг зміненого блоку. Тобто майнінг зміненого блоку для знаходження нового валідного нонса.

Крок 8. Повторна перевірка валідності блокчейну. Тобто після зміни даних перевіряється, чи ланцюг блоків все ще є валідним.

Крок 9. Повернення результату перевірки (True/False).

Після проведення побудови алгоритму програми та алгоритму блокчейну для розроблюваного web-ресурсу для онлайн трейдингу можна зробити висновки, що обрана архітектура та принципи дизайну спрямовані на досягнення балансу між децентралізованою та розподіленою природою технології блокчейн і потребою у створенні зручного та безпечного інтерфейсу для широкого впровадження.

Завдяки безперешкодній інтеграції мережі блокчейн з спроектованим захищеним веб-додатком на основі цієї технології, сховищем даних та іншими компонентами, система забезпечує комплексне рішення, що передбачає основні функції, які передбачає система для трейдингу.

2.3 Розробка бази даних web-ресурсу

Для розробки захищеної бази даних визначеного корпоративного та захищеного web-ресурсу було використано Flask-MySQLdb.

Рішення використовувати дану реляційну базу даних для управління користувачами зумовлене потребою в ефективних запитах, обмеженнями цілісності даних і можливістю встановлювати зв'язки між різними сутностями конкретної бази даних для захищеного веб-ресурсу для онлайн трейдингу (наприклад, користувачами та пов'язаними з ними транзакціями).

Однією з реляційних систем управління базами даних (СКБД), яка доступна безкоштовно, є MySQL. У реляційних базах даних дані впорядковуються в одну або кілька таблиць так, що вони можуть бути пов'язані між собою і стають більш структурованими. Програмісти використовують SQL для управління доступом користувачів до бази даних, а також для створення, редагування та отримання даних з реляційних баз даних. СУБД, такі як MySQL, співпрацюють з операційною системою для впровадження реляційних баз даних у систему зберігання даних комп'ютера, управління користувачами, надання доступу до мережі, а також резервного копіювання та перевірки цілісності баз даних на додаток до реляційних баз даних та SQL [31].

Використовуючи цю систему УБД було використано основні команди DML – мови маніпуляцій базою даних, серед них:

1. Select, що використовується для використання вибірки даних з бази:

```
SELECT * FROM employees.
```

2. Insert, що додає нові записи:

```
INSERT INTO employees (name, position, salary) VALUES ('John', 'Manager', 50000).
```

3. Update, що оновлює існуючі записи:

```
UPDATE employees SET salary = 55000 WHERE name = 'John'.
```

4. Delete, що видаляє записи:

```
DELETE FROM employees WHERE name = 'John'.
```

Також команди DDL, тобто мову опису даних, DCL (мова контролю даних) та TCL, яка є мовою контролю транзакцій.

Зважаючи на той факт, що було використано не просто MySQLdb, а ще й додатковий фреймворк Flask, який є гнучким і легким для реалізації рішенням на основі Python. Мінімалістичний дизайн Flask забезпечує високий ступінь

кастомізації та відосно безперешкодну інтеграцію з іншими, що було додано під час практичної реалізації проекту. Його основою є шаблонізатор Jinja2 та інструменти WSGI. Движок шаблонів Jinja2, який постачається з Flask, дозволяє ефективно генерувати динамічні HTML-шаблони, що в свою чергу допоможе створити більш зручний інтерфейс [32].

Для підвищення безпеки веб-додатку, розробка якого передбачена у даній роботі, було використано бібліотеку PassLib для безпечного хешування та верифікації паролів. Ця бібліотека надає повний набір алгоритмів та утиліт для обробки паролів, солей та інших конфіденційних даних, дотримуючись найкращих галузевих стандартів.

Розширення Flask-MySQLdb забезпечило безперешкодну інтеграцію між додатком Flask і базою даних MySQL (рис. 2.5).

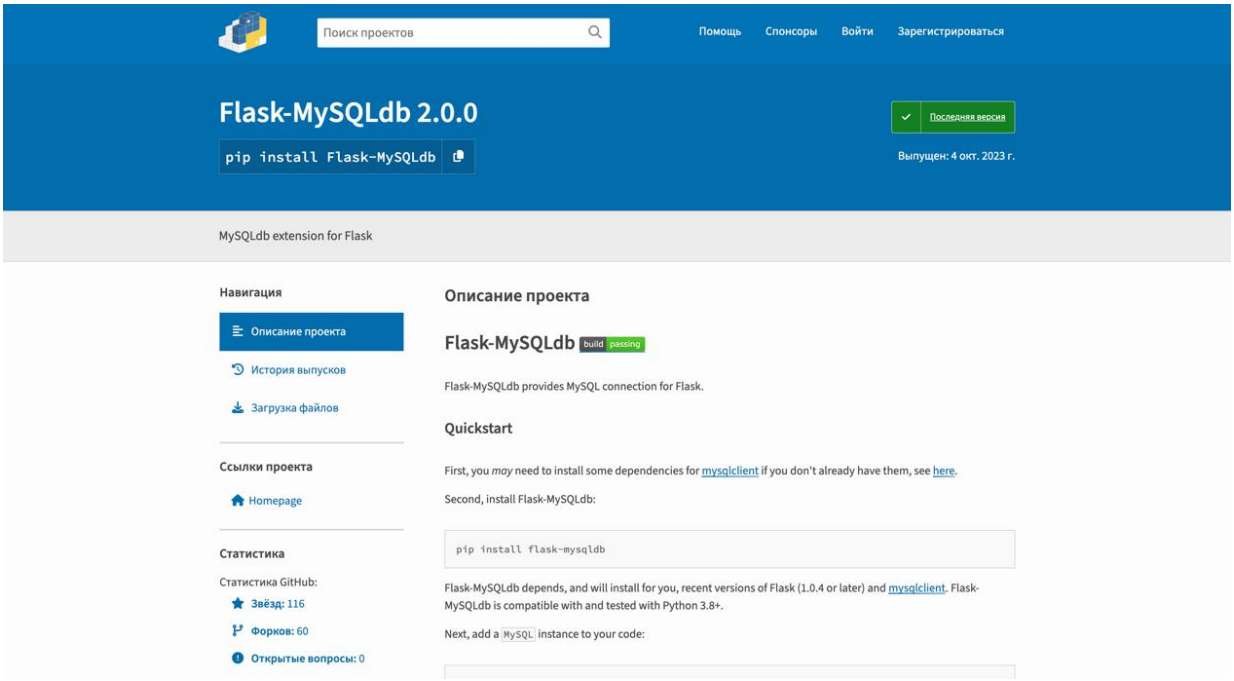


Рисунок 2.5 – Бібліотека Flask-MySQLdb

Це розширення спростило процес виконання SQL-запитів, отримання даних та управління з'єднаннями з базою даних у середовищі Python.

Система управління користувачами, яка обробляє реєстрацію, автентифікацію та авторизацію користувачів, покладається на реляційну систему управління базами даних (RDBMS) для зберігання інформації про користувачів.

Схема бази даних для системи управління користувачами розроблена таким чином, щоб забезпечити цілісність даних, безпеку та ефективні запити для веб-додатку, аби зробити його конкурентоспроможним. Первинною таблицею в схемі є таблиця "users", яка зберігає основну інформацію про користувачів, таку як:

1. Ідентифікатор користувача – id.
2. Ім'я користувача – name.
3. Електронна пошта – email.
4. Внутрішньо платформне ім’я – username.
5. Пароль – password.

Для реалізації такої таблиці було розроблено код виконання:

```
def isnewuser(username):
    users = Table("users", "name", "email", "username", "password")
    data = users.getall()
    usernames = [user['username'] for user in data]
    return False if username in usernames else True
```

Тож схема бази даних для визначеного ресурсу буде виглядати наступним чином (рис. 2.6).

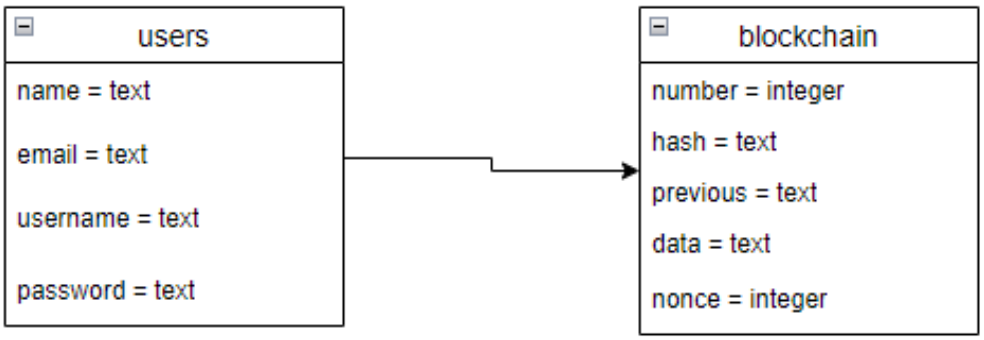


Рисунок 2.6 – Схема бази даних для трейдинг платформи

Проте для реалізації задуманого проекту та розробки повноцінної бази даних окрім вище описаної потрібні й інші сутності: КОРИСТУВАЧІ та БЛОКЧЕЙН.

Ще одною з важливих складових розробки бази даних є її аналіз за допомогою UML-діаграми. Отже, для побудови відповідної діаграми для початку потрібно визначити основні класи, функції та взаємодії.

В цілому для аналізу можна було б застосувати XML-діаграму, проте, зазвичай вони не застосовуються для такого роду кодів, який було спроектовано для даного проекту, тому основною задачею буде саме UML-діаграма (рис. 2.7).

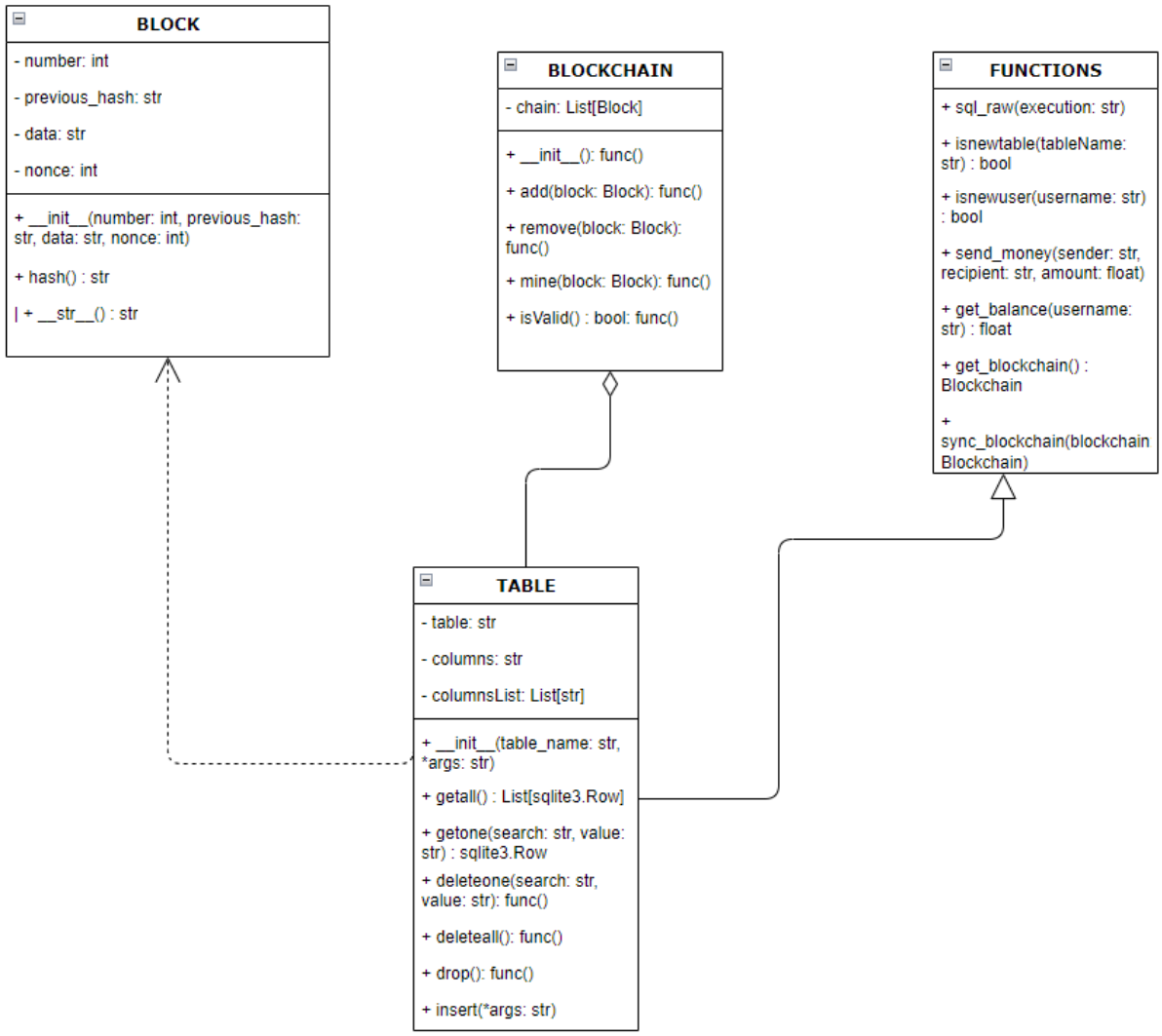


Рисунок 2.7 – Основна UML діаграма класів для web-додатку для трейдингу з використанням blockchain технології

Основними класами для визначеного коду є: Block, Blockchain, Table та деякі основні функції.

При попередній побудові даної діаграми було значно полегшено процес практичної реалізації захищеного веб-ресурсу для онлайн трейдингу, адже розробка такого виду схем дозволяє не заплутуватись під час розробки, враховувати всі чинники та залежності між функціями, класами та модулями, які було використано при проектуванні.

2.4 Висновки до розділу

Під час виконання даного розділу було проведено розробку структури та функціоналу запланованого для розробки захищеного корпоративного web-додатку. Що передбачало основні визначення основних критеріїв та вимог до розробки. А також було описано схему комунікації користувача та розробленого веб-ресурсу.

Цей розділ також містив розробку алгоритму функціонування та алгоритмувпровадження блокчейн технології web-системи для онлайн трейдингу, що були основними складовими при проектуванні, адже технологія блокчейн є базисом захищеності розроблюваного веб-додатку для онлайн трейдингу.

Було проведено планування та розробку бази даних, що зберігатие та оброблятиме дані, у цьому допомогла СУБД MySQL а додаток до Java Script – Flask-MySQLdb. Комбінація цих технологій забезпечить коректну та впевнену роботу бази даних для ресурсу з онлайн трейдингу на основі технології блокчейн.

РОЗДІЛ 3. ОПИС РОЗРОБКИ WEB-РЕСУРСУ ДЛЯ ОНЛАЙН ТОРГІВЛІ

Даний розділ передбачає опис основних етапів практичної реалізації та тестування розробленого захищеного корпоративного web-ресурсу для надання послуг з онлайн торгівлі за допомогою блокчейн технології, також обґрунтування вибору засобів для програмування, що відіграють ключову роль у зручності розробки та якості вихідного коду.

3.1 Вибір засобів програмування для розроблюваного додатку

Проаналізувавши поставлену мету та задачі, що потрібно виконати для успішної реалізації захищеного корпоративного web-ресурсу для надання послуг з онлайн торгівлі за допомогою блокчейн технології обрано такі засоби програмування:

1. Visual Studio Code.
2. Java Script.
3. SCSS.
4. Python.
5. MySQLdb.
6. Flask-MySQLdb.
7. Bootstrap

Для реалізації задуманої задачі існує багато різних ресурсів, проте було обрано найоптимальніший та найроспосвюдженіший варіант – Visual Studio Code (рис. 2.2).

Обране середовище розробки мало на меті досягти балансу між використанням стандартних галузевих технологій та впровадженням передових рішень, пристосованих до унікальних вимог блокчейну та децентралізованих додатків.

Visual Studio Code (також відомий як VS Code) - це міні-версія Visual Studio. Це легкий текстовий редактор з відкритим вихідним кодом, доступний для Windows, Mac і Linux [33].

Його можна використовувати з різними мовами програмування, включаючи C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust та Julia. Visual Studio Code використовує той самий компонент редактора (під кодовою назвою «Monaco»), що й Azure DevOps (раніше називався «Visual Studio Online» та «Visual Studio Team Services»).

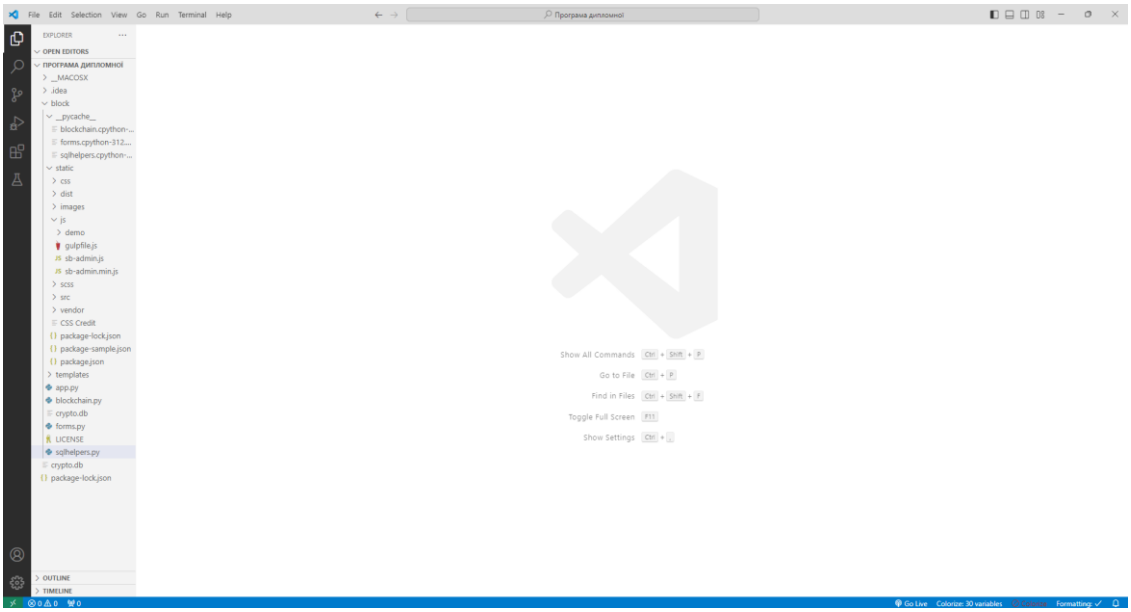


Рисунок 3.1 – Середовище розробки Visual Studio Code

Для наочного прикладу того, що дана програма відповідає всім вимогам, швидко розвивається та є найоптимальнішим рішенням для розробки обраного технічого рішення є статистика відгуків розробників.

До прикладу, в опитуванні розробників 2016 року про переповнення стеку Visual Studio Code посів 13 місце серед найпопулярніших інструментів розробки, лише 7% з 47 000 респондентів використовували його [34]. Через два роки, однак, Visual Studio Code досягнув місця № 1, з 35% з 75 000 респондентів, які його використовували. В опитуванні розробників 2019 року Visual Studio Code також посів місце №. 1, з 50% з 87 000 респондентів, які його використовували [35]. Опитування розробників 2020 року не охоплювало інтегровані середовища розробки [36]. В опитуванні розробників 2021 року Visual Studio Code продовжував займати перше місце, з 74,5% з 71 000 респондентів, які його використовували

[37],74,48% з 71 010 відповідей в опитуванні 2022 року, і 73,71% з 86 544 відповідей в опитуванні 2023 року [38].

В основі системи лежить реалізація мережі блокчейн, для розробки якої було використано мови програмування Python та Java Script.

Даний вибір мов програмування аргументується зрозумілістю та широкою екосистемою бібліотек. Модуль hashlib, що входить до стандартної бібліотеки Python, був використаний для реалізації функцій криптографічного хешування, необхідних для безпечного зберігання та перевірки даних у блокчейні[39].

Python - це універсальна мова програмування високого рівня. З сильним акцентом на відступи, її філософія проектування надає пріоритет читабельності коду. Python використовує збір сміття та динамічну типізацію. Він сумісний з кількома парадигмами програмування, такими як об'єктно-орієнтований, функціональний та структурований [40].

Звичайно, додаток було побудовано з використанням веб-фреймворку Flask, що є рішенням-фреймворком на основі Python. При описі розробки бази даних вже було описано його переваги, до яких входять легкість та зручність у розробці.

Для розробки інтерфейсу веб-додатку було використано комбінацію HTML5, SASS та JavaScript, а фреймворк Bootstrap забезпечив послідовний та візуально привабливий дизайн для різних компонентів. Також доволі важливим компонентом розробки слугувала бібліотека jQuery, що використовується для покращення користувацького досвіду, забезпечуючи беззупинну взаємодію з базою даних зі сторони користувача за допомогою AJAX.

Багатофункціональна, компактна та швидка бібліотека JavaScript називається jQuery. Завдяки простому у використанні API, який працює в декількох браузерах, вона спрощує такі завдання, як обробка подій, анімація, Ajax, а також переміщення та маніпулювання HTML-документами. Мільйони розробників JavaScript створюють різноманітний код завдяки адаптивності та гнучкості jQuery [41].

AJAX у свою чергу полегшує асинхронне оновлення веб-сайтів, обмінюючись даними з веб-сервером за лаштунками. Це означає, що частини веб-сторінки можуть оновлюватися без необхідності перезавантаження сторінки [42].

HTML (мова гіпертекстової розмітки) використовується для структурування змісту і макета веб-сторінок, визначаючи різні елементи, такі як заголовки, абзаци, форми і таблиці, а також вбудовувати на сторінку такий контент, як зображення та відео. Документи HTML відповідають стандарту HTML5, що забезпечує сумісність із сучасними веб-браузерами та надає доступ до нових функцій і API [43].

Каскадні таблиці стилів (CSS) - це мова таблиць стилів, що використовується для визначення презентації та стилю документа, написаного мовою розмітки HTML або XML (включаючи діалекти XML, такі як SVG, MathML або XHTML) [44].

JavaScript спочатку був створений для того, щоб «оживляти» веб-сторінки. Програми на цій мові називаються скриптами. Вони можуть бути написані прямо в HTML-коді веб-сторінки і запускатися автоматично при її завантаженні. Скрипти надаються і виконуються у вигляді звичайного тексту. Вони не потребують спеціальної підготовки або компіляції для запуску. У цьому аспекті JavaScript дуже відрізняється від іншої мови, яка називається Java. Система використовує сучасний JavaScript (ES6+) і популярні бібліотеки та фреймворки, такі як jQuery і Bootstrap [45].

Комбінація вищезгаданих технологій ідеально підходить для реалізації корпоративного захищеного веб-додатку для онлайн торгівлі, адже вона має усі можливості у візуальних аспектах, тож забезпечить максимальний комфорт користувача при користуванні розробленим продуктом.

Дані в різних форматах, у тому числі у вигляді JSON та URL-коду, пересилаються між зовнішнім і внутрішнім компонентами системи. JSON - це загальновживаний формат для надсилання структурованих даних через HTTP, оскільки він є читабельним, легким і простим для сприйняття клієнтськими та серверними компонентами.

URL-кодовані дані, які представляють собою пари ключ-значення у певному форматі (наприклад, key1=value1&key2=value2), зазвичай використовуються для передачі даних форм або простих параметрів запитів у HTTP-запитах (рис. 3.2).

У процесі розробки контроль версій відігравав вирішальну роль у забезпеченні ефективної співпраці, організації коду та відстежуваності. Було

використано розподілену систему контролю версій Git з віддаленим репозиторієм, розміщеним на платформі GitHub або GitLab. Таке налаштування полегшило обмін кодом, розгалуження, злиття та відстеження змін серед членів команди розробників.

Для оптимізації процесів розробки та розгортання були використані технології контейнеризації. Для пакування додатку та його залежностей у легкі, портативні контейнери було використано Docker, широко розповсюджену контейнерну платформу. Такий підхід забезпечив послідовне та відтворюване розгортання в різних середовищах, мінімізувавши проблеми сумісності та полегшивши масштабування.

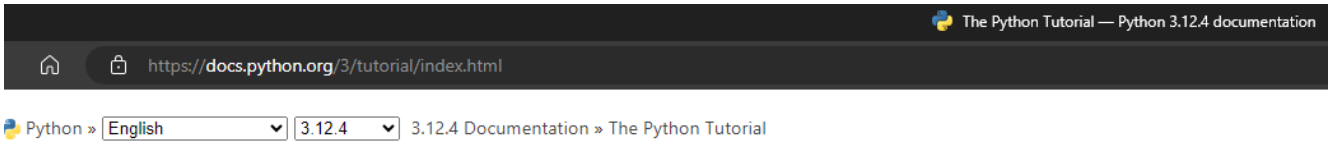


Рисунок 3.2 – Приклад кодованих даних URL

Docker - це набір продуктів платформи як послуги (PaaS), які використовують віртуалізацію на рівні операційної системи для доставки програмного забезпечення в пакетах, що називаються контейнерами [46].

Крім того, середовище розробки включало інструменти для лінтування коду та статичного аналізу, такі як Pylint і Pyflakes для Python та ESLint для JavaScript. Ці інструменти допомогли забезпечити дотримання стандартів кодування, виявити потенційні проблеми в коді та підтримувати узгоджену кодову базу в рамках проекту.

Отже, підсумовуючи всі дані, наведені при формуванні бази ресурсів, що було використано у розробці захищеного веб-ресурсу для онлайн трейдингу з використанням блокчейн технології, можна стверджувати, що така варіативність та поєднання мов програмування, фреймворків та додатків для кодингу забезпечує максимальну зручність при розробці, а також змогу застосовувати усі переваги сучасний основ розробки.

3.2 Розробка програмної частини захищеного web-додатку

Даний підрозділ передбачає представлення основних кроків при розробці проаналізованого, попередньо спроектованого та визначеного за темою додатку.

Звичайно, для початку розробки потрібно було створити основний файл програми, за допомогою якого можна буде проводити запуск додатку та який буде відігравати ключову роль у комунікації між іншими складовими проекту.

Тож, було створено файл «app.py» та розроблено початкову логіку.

Для початку було створено екземпляр Flask-додатку:

```
app = Flask(__name__);
```

Далі проведено підключення до бази даних з використанням бібліотеки «sqlite3»:

```
conn = sqlite3.connect('crypto.db', check_same_thread=False)
conn.row_factory = sqlite3.Row
```

Описано функцію, що забезпечить отримання підключення до бази даних з боку фактичного потоку:

```
def get_db():
    if not hasattr(_thread_local, 'conn'):
        _thread_local.conn = sqlite3.connect('crypto.db')
    return _thread_local.conn
```

Наступною дуже важливою функцією є перевірка на стан користувача, тобто чи вже увійшов у систему:

```
def is_logged_in(f):
    @wraps(f)
    def wrap(*args, **kwargs):
        if 'logged_in' in session:
            return f(*args, **kwargs)
        else:
            flash("Unauthorized, please login.", "danger")
            return redirect(url_for('login'))
    return wrap
```

Функція входження користувача, що забезпечує оновлення сесії виглядає наступним чином:

```
def log_in_user(username):
    users = Table("users", "name", "email", "username", "password")
    user = users.getone("username", username)

    session['logged_in'] = True
    session['username'] = username
    session['name'] = user['name']
    session['email'] = user['email']
```

Для коректної роботу додатку буде прописано сторінку реєстрації, що забезпечить початкову функціональність додатку, продукує реєстрацію та перенесення даних реєстрації в основну таблицю:

```
@app.route("/register", methods = ['GET', 'POST'])
def register():
    form = RegisterForm(request.form)
    users = Table("users", "name", "email", "username", "password")
```

Згодом проводиться перевірка на те, що форма завершена і дані від користувача зможуть надійти до бази даних:

```
if request.method == 'POST' and form.validate():
username = form.username.data
email = form.email.data
name = form.name.data
```

Також важливою є перевірка на те, чи користувач вже не існував раніше, адже відсутність такої перевірки може значно вплинути на завантаженість системи:

```
if isnewuser(username):
password = sha256_crypt.encrypt(form.password.data)
            users.insert(name,email,username,password)
            log_in_user(username)
            return redirect(url_for('dashboard'))
        else:
            flash('User already exists', 'danger')
```

```
        return redirect(url_for('register'))
```

Звичайно, сторінка реєстрації не єдине, що потрібно було розробити, тому наступним кроком стала розробка сторінки входження, тобто потрібна користувачам, що вже раніше проходили реєстрацію.

Таким самим чином проводиться перевірка на те, що форма передала інформацію базі даних:

```
if request.method == 'POST':
    # Collect form data
    username = request.form['username']
    candidate = request.form['password']
```

Прописано звернення до таблиці користувачів, для того, щоб отримати пароль:

```
users = Table("users", "name", "email", "username", "password")
    user = users.getone("username", username)
```

Важливо було перевірити, чи введені дані взагалі співпадають з вже створеними раніше, тож відбувається перевірка на існування користувача:

```
if user is None:
        flash("Username is not found", 'danger')
        return redirect(url_for('login'))
    else:
```

Далі, перевірено правильність паролю:

```
accPass = user.get('password')
        if sha256_crypt.verify(candidate, accPass):
```

А після успішного проходження перевірок виконання перенесення користувача на сторінку з розширеними можливостями:

```
log_in_user(username)
            flash('You are now logged in.', 'success')
            return redirect(url_for('dashboard'))
        else:
```

Якщо ж перевірка не була пройдено, то повернення на сторінку входження:

```
flash("Invalid password", 'danger')
```

```
            return redirect(url_for('login'))
```

Сторінка транзакцій вже відноситься до внутрішній функцій плфторми для трейдингу і також потребує початкової ініціалізації:

```
@app.route("/transaction", methods = ['GET', 'POST'])
@is_logged_in
def transaction():
  form = SendMoneyForm(request.form)
  balance = get_balance(session.get('username'))
```

Після цього проводиться перевірка на підтвердження операції та переведення коштів:

```
if request.method == 'POST':
    try:
      send_money(session.get('username'), form.username.data, form.amount.data)
      flash("Money Sent!", "success")
    except Exception as e:
      flash(str(e), 'danger')
    return redirect(url_for('transaction'))
  return render_template('transaction.html', balance=balance, form=form, page='transaction')
```

Останнім основним елементом базового фалу є створення сторінки для покупки валюти, що також передбачає ініціалізацію сторінки та створення логіки для переведення коштів:

```
@app.route("/buy", methods = ['GET', 'POST'])
@is_logged_in
def buy():
  form = BuyForm(request.form)
  balance = get_balance(session.get('username'))
if request.method == 'POST':
    try:
      send_money("BANK", session.get('username'), form.amount.data)
      flash("Purchase Successful!", "success")
    except Exception as e:
      flash(str(e), 'danger')
```

```
return redirect(url_for('dashboard'))
```

Звичайно, основною темою є створення трейдингової платформи, проте зазначено використання блокчейн технології, тож, було створено файл «blockchain.py», який передбачає наявність логіки технології блокчейн.

Для початку, у створеному файлі потрібно отримати дані з біліотеки «hashlib» для отримання функцій хешування SHA256:

```
from hashlib import sha256
```

Наступний відрізок коду передбачає повернення хеш коду з отриманого переліку аргументів:

```
def updatehash(*args):
    hashing_text = ""; h = sha256()
    for arg in args:
        hashing_text += str(arg)
    h.update(hashing_text.encode('utf-8'))
    return h.hexdigest()
```

Створення основних та раніше спроектованих класів «Block» та «Blockcain».

Для основної програмної логіки першого, було створено початкову інформацію для ще неопрацьованого блоку:

```
def __init__(self,number=0, previous_hash="0"*64, data=None, nonce=0):
        self.data = data
        self.number = number
        self.previous_hash = previous_hash
        self.nonce = nonce
```

Далі проводиться повернення хеш блоку з отриманих даних:

```
def hash(self):
        return updatehash(
            self.number,
            self.previous_hash,
            self.data,
            self.nonce
        )
```

І в кінці повертає стрічку блоків з даними:

```
def __str__(self):
    return str("Block#: %s\nHash: %s\nPrevious: %s\nData: %s\nNonce: %s\n" %(
      self.number,
      self.hash(),
      self.previous_hash,
      self.data,
      self.nonce
      )
    )
```

Після створення визначеного класу відбувається побудова класу, що вже відповідає саме за ланцюг блоків, а не за кожен блок окремо, тож для початку визначається складність обробки:

```
difficulty = 4
```

Перезапуск ланцюга вже існуючого або нового під час повторної ініціалізації:

```
def __init__(self):
    self.chain = []
```

Додавання нового блоку до ланцюга:

```
def add(self, block):
    self.chain.append(block)
```

І також видалення як обернена функція:

```
def remove(self, block):
    self.chain.remove(block)
```

Далі відбувається знаходження поняття складності блоку для допуску до загального ланцюга:

```
def mine(self, block):
    try: block.previous_hash = self.chain[-1].hash()
    except IndexError: pass
```

І відповідно в цьому процесі відбувається пошук до моменту знаходження:

```
while True:
      if block.hash()[:self.difficulty] == "0" * self.difficulty:
```

```
            self.add(block); break
        else:
            block.nonce += 1
```

Важливим етапом стала перевірка блокчейну на валідність:

```
def isValid(self):
    for i in range(1,len(self.chain)):
        _previous = self.chain[i].previous_hash
        _current = self.chain[i-1].hash()
        if _previous != _current or _current[:self.difficulty] != "0"*self.difficulty:
            return False
    return True
```

Далі була створена основна функція, що передбачає цілісне виконання прописаного коду, його валідацію та як зміна даних впливає на валідацію:

```
def main():
    blockchain = Blockchain()
    database = ["hello", "goodbye", "test", "DATA here"]
    num = 0
    for data in database:
        num += 1
        blockchain.mine(Block(num, data=data))
    for block in blockchain.chain:
        print(block)
    print(blockchain.isValid())
    blockchain.chain[2].data = "NEW DATA"
    blockchain.mine(blockchain.chain[2])
    print(blockchain.isValid())
if __name__ == '__main__':
    main()
```

Для управління базою даних та коректної роботи таблиць було створено файл «sqlhelpers.py», що передбачає процедури створення таблиць та маніпуляції даними.

Створення зв’язку з базою даних:

```
conn = sqlite3.connect('crypto.db', check_same_thread=False)
conn.row_factory = sqlite3.Row
```

Введення кастомних дозволів для помилок транзакцій:

```
class InvalidTransactionException(Exception): pass
class InsufficientFundsException(Exception): pass
```

При створенні таблиці відбувається перевірка на її існування:

```
if isnewtable(table_name):
        create_data = ",".join([f"{column} TEXT" for column in self.columnsList])
        cur = conn.cursor()
        cur.execute(f"CREATE TABLE {self.table}({create_data})")
        conn.commit()
        cur.close()
```

Функція отримання усіх даних з таблиці:

```
def getall(self):
    cur = conn.cursor()
    result = cur.execute(f"SELECT * FROM {self.table}")
    data = result.fetchall()
    cur.close()
    return data
```

Отримання одного значення з таблиці, базуючись на даних по стовпцях:

```
def getone(self, search, value):
    data = {}
    cur = conn.cursor()
    result = cur.execute(f"SELECT * FROM {self.table} WHERE {search} = ?", (value,))
    data = result.fetchone() if result else None
    cur.close()
    return data
```

Відповідно видалення даних за тим же принципом:

```
def deleteone(self, search, value):
    cur = conn.cursor()
    cur.execute(f"DELETE FROM {self.table} WHERE {search} = ?", (value,))
    conn.commit()
```

```
        cur.close()
```

Видалення усіх даних із таблиці:

```
def deleteall(self):
        self.drop()
        self.__init__(self.table, *self.columnsList)
```

Код передбачає ще кілька побічних операцій, проте приділено увагу перевірці на наявність користувача в наявній базі:

```
def isnewuser(username):
    users = Table("users", "name", "email", "username", "password")
    data = users.getall()
    usernames = [user['username'] for user in data]


    return False if username in usernames else True
```

А також важливим відрізком коду є передання від одного користувача до іншого:

```
def send_money(sender, recipient, amount
    try: amount = float(amount)
    except ValueError:
        raise InvalidTransactionException("Invalid Transaction")
    if amount > get_balance(sender) and sender != "BANK":
        raise InsufficientFundsException("Insufficient Funds.")
    elif sender == recipient or amount <= 0.00:
        raise InvalidTransactionException("Invalid Transaction.")
    elif isnewuser(recipient):
        raise InvalidTransactionException("User Does Not Exist.")
    blockchain = get_blockchain()
    number = len(blockchain.chain) + 1
    data = f"{sender}-->{recipient}-->{float(amount)}"
    blockchain.mine(Block(number, data=data))
    sync_blockchain(blockchain)
```

Написання вище описаного коду, тобто класів відношень, перевірок та функцій передбачає створення основної логіки програми. Проте, практична

реалізація також містила виконання програмування мовою JavaScript та її фреймворками для візуалізації описаної логіки та надання додатку привабливого візуального стану. Для цього також було використано мову гіпертекстової розмітки – HTML і, звичайно, для стилізації було використано CSS та SASS.

3.3. Тестування розроблено програмного засобу для онлайн трейдингу

Інтерфейс користувача (UI) криптовалютної системи, заснованої на блокчейні, є одною з найважливіших складових розробки, адже переважно користувача не хвилює те, що відбувається в кодовій частині, лише факт безпеки, а в цілому користувачу важливіше візуальне представлення програмного ресурсу. Інтерфейс розроблений таким чином, щоб користувачі могли безперешкодно та легко взаємодіяти з децентралізованою фінансовою екосистемою (рис. 3.3).

Основна сторінка, на якій відображається статистика визначеної криптовалюти та історія транзакцій є візуально привабливою на вигляд, адже не передбачає нагромадження інформації. Дана сторінка надає чіткі інструкції для нових користувачів, щоб зареєструватися, і для існуючих користувачів, щоб увійти в систему, зберігаючи при цьому інформативний дизайн.

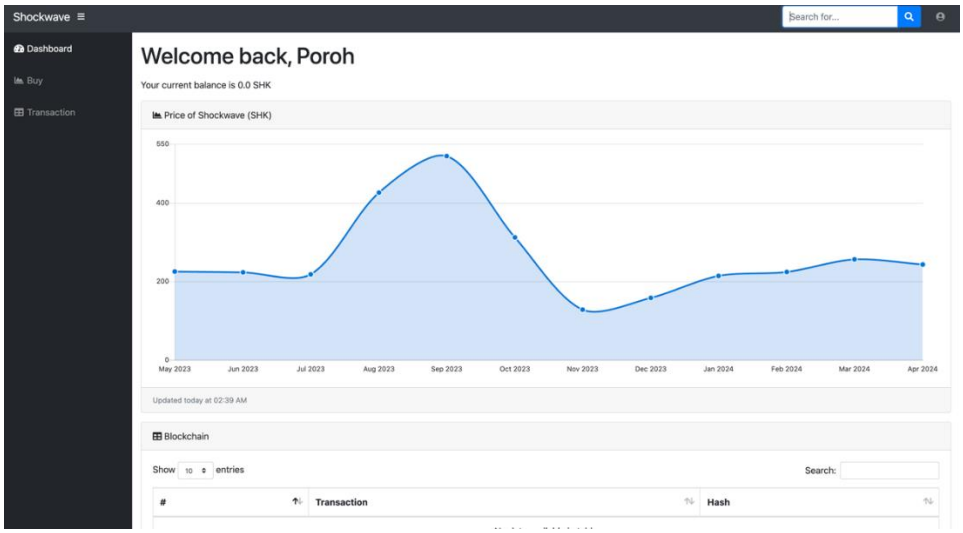


Рисунок 3.3 – Головна сторінка розробленого web-ресурсу

Для доступу до вище згаданої вкладки потрібне створення облікового запису, яке забезпечується простим та зручним процесом реєстрації із приємним

інтерфейсом. У реєстраційній формі використані стандарти дизайну форм, які передбачають чіткі позначення, інструкції для пришвидшення процесу, а також надає фактичну перевірку даних у реальному часі, щоб перевірити правильність уведення даних (рис. 3.4).

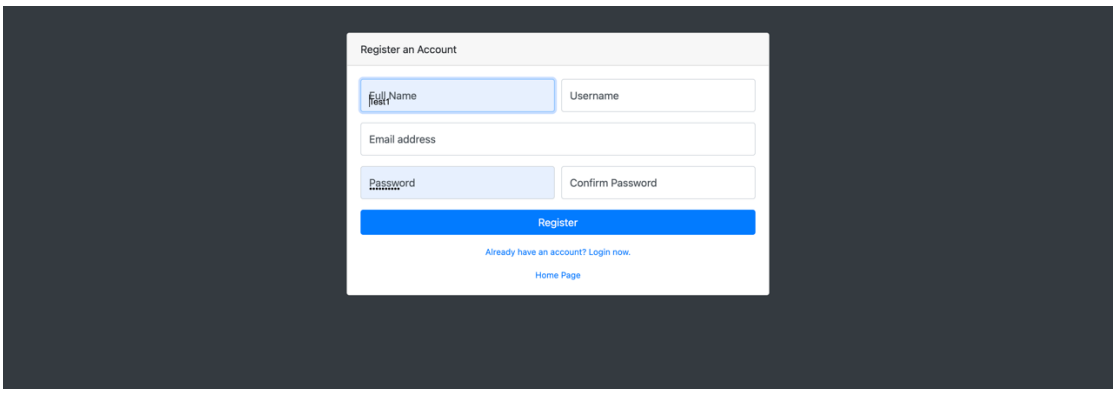


Рисунок 3.4 – Реєстрація користувача у розробленому додатку

Реєстраційна форма дотримується найкращих стандартів дизайну форм, використовуючи чіткі позначення, корисні інструкції та перевірку даних у режимі реального часу, щоб переконатися, що дані введені правильно. Для підвищення безпеки системи введено надійну політику паролів, яка вимагає від користувачів створювати надійні та унікальні паролі відповідно до галузевих стандартів складності.

Проте, платформа передбачає не тільки реєстрацію, а й логінізацію, якщо у користувача уже є свій акаунт і йому не потрібне та незручне створення нового (рис. 3.5).

І вже після того, як користувач успішно зареєструвався додаток автоматично переводить його на головну сторінку, яку вже було представлено вище, також можна описати її як інформаційну панель, що передбачає наявність основної інформації.

Ця панель служить основним місцем для керування своїми криптовалютними гаманцями, відстежування транзакцій та отримування доступ до різних функцій системи. Панель управління розроблена з використанням адаптивного макета, який легко адаптується до різних розмірів і роздільної здатності екранів.

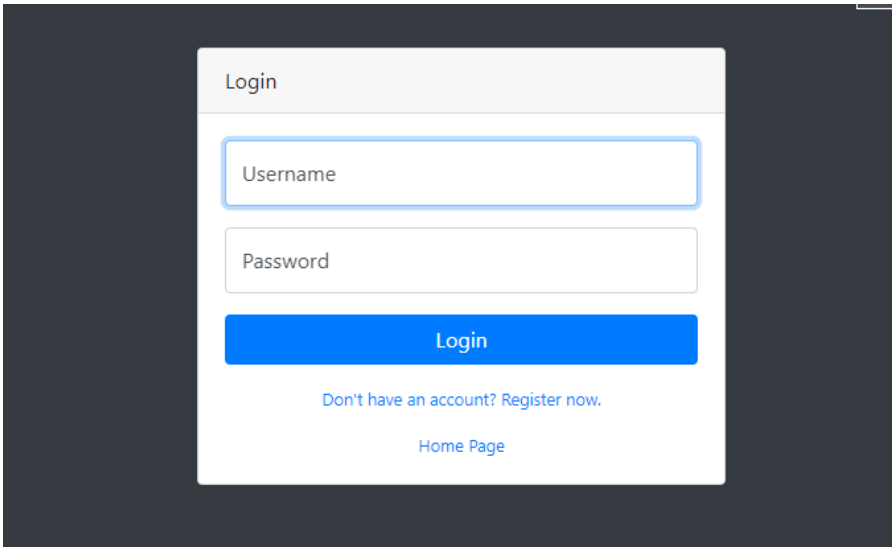


Рисунок 3.5 – Вікно логінізації користувача у ресурс для онлайн трейдингу

Макет програмного додатку був детально розроблений для того, щоб користувачу, комфорт використання ресурсу був на найвищому рівні, тож, з лівого боку було розміщено панель навігації по додатку (рис. 3.6).

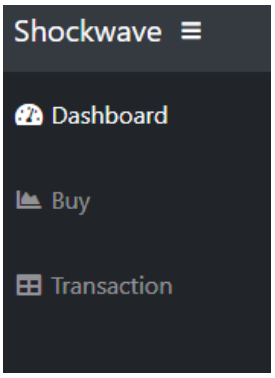


Рисунок 3.6 – Навігаційна панель додатку для онлайн трейдингу

Для зручності додаток має послідовну візуальну ієрархію, де на найвидніших місцях відображаються основні дії та інформація, тоді як другорядні або рідше використовувані функції організовані логічним чином і легко доступні у вже вище згаданій панелі навігації.

Користувачі можуть легко створювати та керувати кількома криптовалютними гаманцями, використовуючи розділ управління гаманцями на інформаційній панелі. Також вони можуть легко переглядати поточні баланси та

історію транзакцій, а для поповнення криптовалютного гаманця потрібно звернутись до вкладки покупки (рис. 3.7).

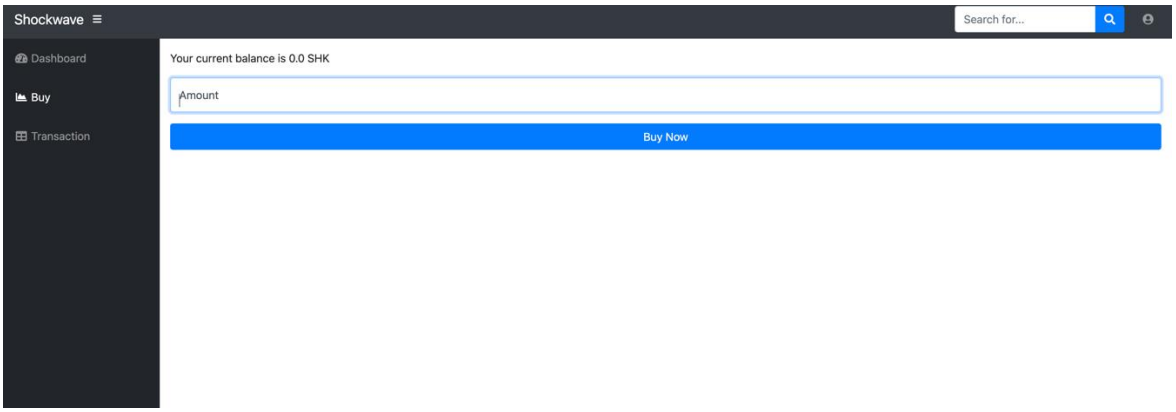


Рисунок 3.7 – Вікно покупки внутрішньої валюти розробленого додатку

Також, варто звернути увагу на вкладку транзакції, що надає змогу користувачу передавати валютні активи іншим користувачам платформи, або на свої ж гаманці (рис.3.8).

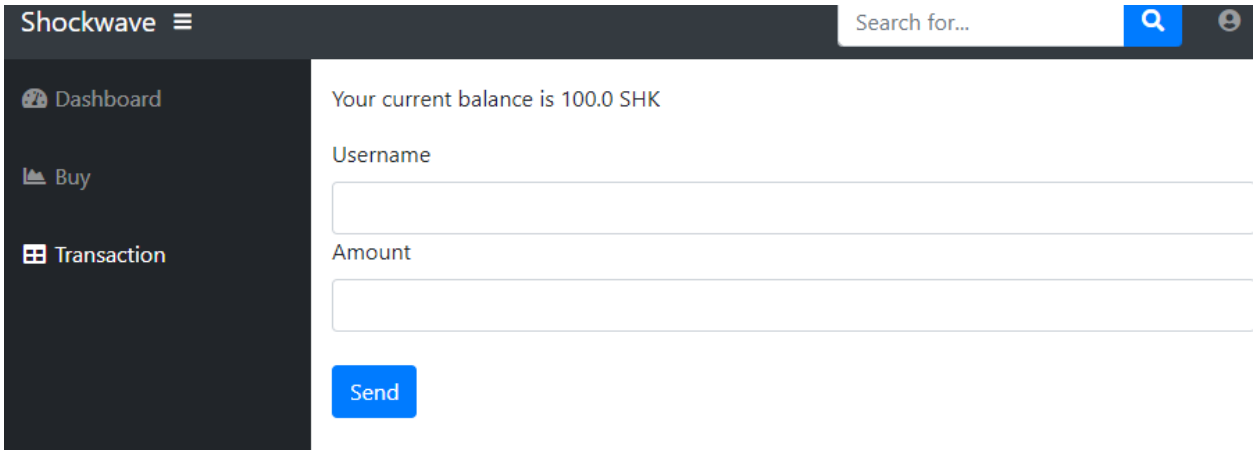


Рисунок 3.8 – Вікно передачі валюти іншим користувачам розробленого ресурсу

Звичайно, інтерфейс передачі містить основні поля, тобто ім’я користувача, якому буде передаватись валюта, а також сума транзакції.

Як вже було згадано вишче, для кращого розуміння користувачів інтерфейс системи включає інтерактивні візуалізації та діаграми (рис. 3.9).

Користувачі можуть бачити графіки в реальному часі з історичними даними про ціни, підтвердження блоків і обсяги транзакцій, що дозволяє їм приймати обґрунтовані рішення щодо своїх інвестицій і транзакцій у криптовалюті.

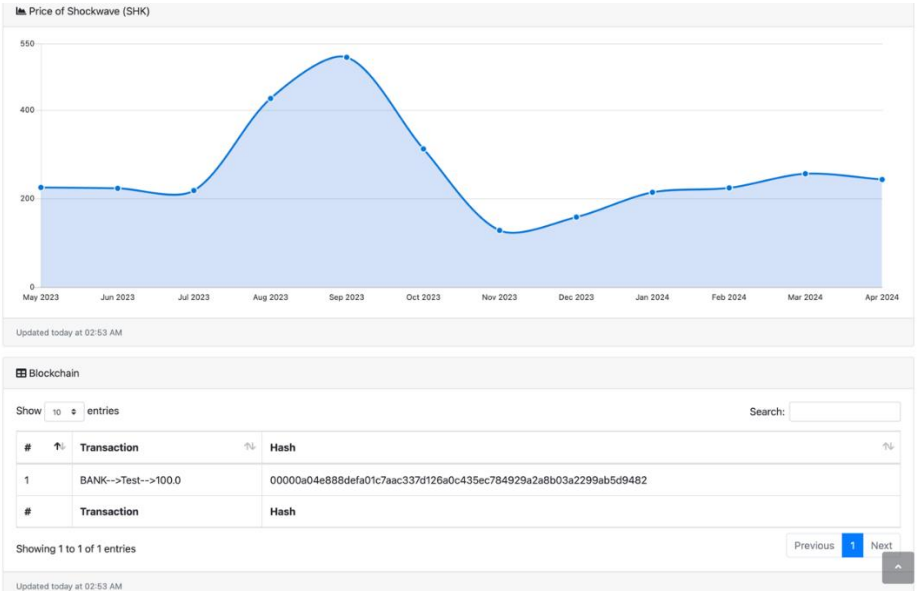


Рисунок 3.9 – Діаграма залежності криптовалюти, що представлена у розробленому ресурсі

Також важливим складником є історія транзакцій, яке надається кожному користувачу платформи для особистої інформованості та зручності (рис. 3.10).

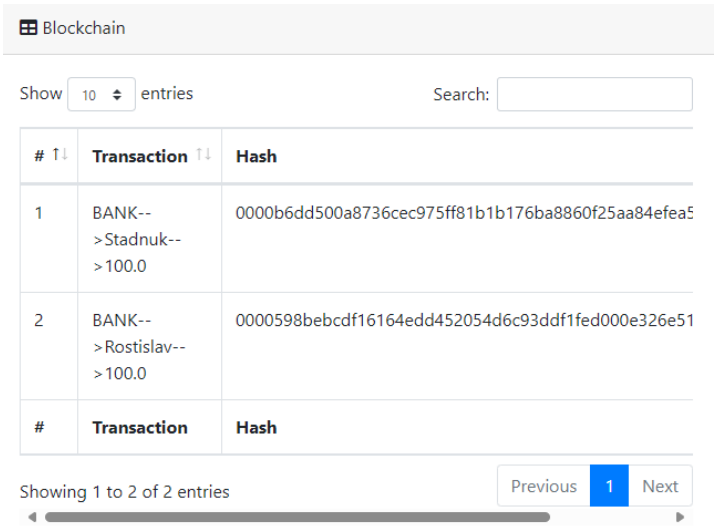


Рисунок 3.10 – Історія транзакцій користувача проведених з використанням створеного ресурсу

Отже, у підсумку даного розділу можна зазначити, що було створено трейдингову платформу з використанням технології блокчейн, що надає змогу користувачам зручно та ефективно оперувати фінансами у своєму гаманці, а також проводити різного аналізи та спостерігати за рухом валюти. Зважаючи на візуальне представлення ресурсу, можна стверджувати, що він є достойним представленням платформи для трейдингу у візуальному аспекті також.

3.4 Висновки до розділу

Під час виконання третього і підсумкового розділу даної роботи було проведено аргументацію та опис прийнятих рішень, що до програмування web-ресурсу. Такими засобами стали: Visual Studio Code, Java Script, SCSS, Python, MySQLdb, Flask-MySQLdb, Bootstrap.

Фактичною основою виконання даного розділу було впровадження програмних рішень, що були спроектовані і сформовані у попередньому розділі, тож було представлено уривки коду, що передбачали основну логіку та забезпечення функціональності системи.

Звичайно, у даному розділі також було представлено візуальний інтерфейс додатку, що, також передбачає паралельне тестування, яке містило перевірку всіх функцій та операції, що передбачаються розробленим веб-ресурсом для онлайн трейдингу.

ВИСНОВКИ

Викоання даної бакалаврської роботи мало мету дослідження, тож, воно дозволило розробити і впровадити захищену корпоративну криптовалютну систему на основі блокчейну, яка є діючою і виконує основні функції для трейдингу. Завдяки ретельному аналізу, розробці та впровадженню система продемонструвала свою здатність використовувати технологію блокчейн, що дозволяє безперешкодно здійснювати транзакції криптовалюти та зміцнює довіру до децентралізованої фінансової екосистеми.

Важливим аспектом написання був правильний підхід до створення описаного web-додатку для онлайн трейдингу з використанням технології блокчейн, адже важливим є проектування, планування та фіксація на покроковому та оптимальному створенні. Ознайомлення з теоретичними відомостями, даними про блочейн, а також огляд аналогів дали змогу уникнути багатьох помилок. Другий розділ, що передбачав попередню розробку, огляд алгоритмів та створення діагарм для кращого структурування даних, а також підстав для початку роботи у впроваджені практичного застосування був один із стейкхолдерів даної роботи. І, звичайно, останній розділ, що передбачав аргументацію методів для програмування і фактичну реалізації додатку став підсумковим та найночнішим прикладом успішної реалізації.

Завдяки даній розробці було отримано поширені знання у сфері блокчейну, web-програмування та інших складових, що, звичайно надає змогу розвиватись у майбутньому та проводити дослідження, які можуть бути спрямовані на впровадження передових рішень, щодо масштабування, оптимізації енергоефективності, створення надійних і зручних платформ і навіть отримізації, збільшенню функціональності для вже розробленого додатку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Блокчейн. Binance Academy. URL: Що таке блокчейн і як він працює? | Binance Academy (дата звернення: 05.03.2024).

2. Blockchain. Pinterest. URL: Blockchain - Wikipedia (pinterest.com) (дата звернення 17.03.2024).

3. C. Napoli, G. Pappalardo, E. Tramontana, A math-ematical model for file fragment diffusion and aneural predictor to manage priority queues overbittorrent, International Journal of Applied Math-ematics and Computer Science 26 (2016) 147–160.

4. M. O'Dair, Blockchain: The Internetof Value, Springer International Pub-lishing, Cham, 2019, pp. 15–30.

5. SHA-256 Secure Hash Crypto Engine. CAST. URL: SHA-256 | 256-bit SHA Secure Hash Crypto Engine IP Core (cast-inc.com) (дата звернення 17.03.2024).

6. What is ECDSA Encryption? How does it work? Encryption Counsulting. URL: What is ECDSA Encryption? How Does It Work? (encryptionconsulting.com) (дата звернення 07.04.2024).

7. Milutinović, Monia (2018). "Cryptocurrency". Ekonomika. 64 (1): 105–122. doi:10.5937/ekonomika1801105M. ISSN 0350-137X.

8. Pernice, Ingolf G. A.; Scott, Brett (20 May 2021). "Cryptocurrency". Internet Policy Review. 10 (2). doi:10.14763/2021.2.1561. ISSN 2197-6775.

9. Bezek, Ian (14 July 2021). "What Is Proof-of-Stake, and Why Is Ethereum Adopting It?".

10. Francesca Fallucchi, Marco Gerardi. Blockchain, State-of-the-Art and Future Trends. Rome, Italy, pp. 80-82.

11. What Is Blockchain? The Tech Behind Crypto Explained. Gemini. URL: What Is a Blockchain Network? Crypto and Beyond | Gemini (дата звернення 06.03.2024).

12. Russo, Camila. "Technology Meant to Make Bitcoin Money Again Is Now Live". March 15, 2018.

13. Editorial: Economic, social and political impacts of blockchain. Science Direct. URL: Editorial: Economic, social and political impacts of blockchain - ScienceDirect (дата звернення 17.04.2024).

14. Blockchain technology in supply chain management for sustainable performance: Evidence from the airport industry. Science Direct. URL: Blockchain technology in supply chain management for sustainable performance: Evidence from the airport industry - ScienceDirect (дата звернення 18.03.2024).

15. Hazari, Shihab S.; Mahmoud, Qusay H. (2019). "Comparative evaluation of consensus mechanisms in cryptocurrencies". Internet Technology Letters.

16. EU Blockchain Observatory and Forum publishes report on Energy Efficiency of Blockchain Technologies. European Commissin. URL: EU Blockchain Observatory and Forum publishes report on Energy Efficiency of Blockchain Technologies | Shaping Europe's digital future (europa.eu) (дата звернення 10.04.2024).

17. The Advantages of Blockchain over Traditional Payments. Nasdaq. URL: The Advantages of Blockchain over Traditional Payments | Nasdaq (дата звернення 15.04.2024).

18. A privacy-preserving scheme with multi-level regulation compliance for blockchain. Scientific reports. URL: A privacy-preserving scheme with multi-level regulation compliance for blockchain | Scientific Reports (nature.com) (дата звернення 17.03.2024).

19. Mona Flink. International blockchain regulation: regulation by code – outlaws or new conceptions of law? Master's Thesis. Advanced International Law and Technology. University of Turku. Faculty of Law. April 2021.

20. BitcoinCore. URL: Bitcoin Core :: About (дата звернення 27.03.2024).

21. Ethereum. URL: What is Ethereum? | ethereum.org (дата звернення 29.03.2024).

22. What is Ethereum and how does it works? Investopedia. URL: What Is Ethereum and How Does It Work? (investopedia.com) (дата звернення 21.04.2024).

23. Gasper. Ethereum. URL: Gasper | ethereum.org (дата звернення 29.03.2024).

24. Peck, Morgan (January 14, 2013). "Ripple Could Help or Harm Bitcoin". IEEE Spectrum. Institute of Electrical and Electronics Engineers.

25. Stellar. URL: Stellar | A Blockchain Network for Payments and Tokenization (дата звернення 20.04.2024).

26. "Coinbase Expands Institutional Services With Tagomi Purchase". Bloomberg News. May 27, 2020.

25. "Coinbase aims to support inclusion with a remote-first workforce". HR Dive. March 17, 2021.

27. Scalability of blockchain: a comprehensive review and future research direction. Springer Link. URL: Scalability of blockchain: a comprehensive review and future research direction | Cluster Computing (springer.com) (дата звернення 27.03.2024).

28. Blockchain Cryptography: Everything You Need to Know. 101 Blockchains. URL: Blockchain Cryptography: Everything You Need to Know - 101 Blockchains (дата звернення 31.04.2024).

29. Jakobsson, Markus; Juels, Ari (1999). "Proofs of Work and Bread Pudding Protocols". Secure Information Networks: Communications and Multimedia Security. Kluwer Academic Publishers: 258–272.

30. What is the Lightning Network? Coinbase. URL: What is the Lightning Network? | Coinbase (дата звернення 17.04.2024).

31. Documentation. MySQL. URL: MySQL :: MySQL 8.4 Release Notes :: Changes in MySQL 8.4.0 (2024-04-30, LTS Release) (дата звернення 16.04.2024).

32. Flask Tutorial. Geeksforgeeks. URL: Flask Tutorial - GeeksforGeeks (дата звернення 17.04.2024).

33. Visual Studio vs Visual Studio Code – What's The Difference Between These IDE Code Editors? freeCodeCamp. URL: Visual Studio vs Visual Studio Code – What's The Difference Between These IDE Code Editors? (freecodecamp.org) (дата звернення 27.04.2024).

34. "Developer Survey Results 2016". Stack Overflow Insights. Stack Exchange.

35. "Developer Survey Results 2019 – Most Popular Development Environments". Stack Overflow Insights. Stack Exchange.

36. "Stack Overflow Developer Survey 2020 - Development Environments and Tools". Stack Overflow Insights. Stack Exchange.

37. "Stack Overflow Developer Survey 2021 - Integrated Development Environment". Stack Overflow Insights. Stack Exchange.

38. "Stack Overflow Developer Survey 2023 - Integrated development environment". Stack Overflow Insights. Stack Exchange.

39. Guido van Rossum, Python Reference Manual, release 2.4.4, 18 October 2006.

40. General Python FAQ. Python. URL: General Python FAQ — Python 3.12.3 documentation (дата звернення 29.03.2024).

41. What is jQuery? jQuery. URL: jQuery (дата звернення 22.04.2024).

42. Ajax introduction. W3schools. URL: AJAX Introduction (w3schools.com) (дата звернення 21.04.2024).

43. Introduction to HTML. Mozilla. URL: Introduction to HTML - Learn web development | MDN (mozilla.org) (дата звернення 28.04.2024).

44. What is CSS? Mozilla. URL: What is CSS? - Learn web development | MDN (mozilla.org) (дата звернення 11.04.2024).

45. What is JavaScript. Mozilla. URL: What is JavaScript? - Learn web development | MDN (mozilla.org) (дата звернення 11.04.2024).

46. O'Gara, Maureen (July 26, 2013). "Ben Golub, Who Sold Gluster to Red Hat, Now Running dotCloud". SYS-CON Media.

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції "Управління інформаційною безпекою" кафедри МБІС
д.т.н., професор
_____ Юрій ЯРЕМЧУК
" 22 " березня 2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до бакалаврської дипломної роботи на тему:

Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології

08-72.БДР.023.00.063.ТЗ

Керівник бакалаврської дипломної роботи

к.т.н., доцент кафедри МБІС
_____ Грицак А. В.
" 22 " березня

Вінниця – 2024 р.

1. Найменування та область застосування

Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології. Область застосування: захист веб-ресурсів для онлайн-торгівлі від несанкціонованого доступу.

2. Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № 80 від 11.03.2024 р.

3. Мета та призначення розробки

3.1 Мета розробки: розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології.

3.2 Призначення: розроблений програмний засіб надає послуги з онлайн-торгівлі за допомогою блокчейн-технології.

4. Джерела розробки

4.1. C. Napoli, G. Pappalardo, E. Tramontana, A math-ematical model for file fragment diffusion and aneural predictor to manage priority queues overbittorrent, International Journal of Applied Math-ematics and Computer Science 26 (2016) 147–160.

4.2. Jakobsson, Markus; Juels, Ari (1999). "Proofs of Work and Bread Pudding Protocols". Secure Information Networks: Communications and Multimedia Security. Kluwer Academic Publishers: 258–272.

4.3. Pernice, Ingolf G. A.; Scott, Brett (20 May 2021). "Cryptocurrency". Internet Policy Review. 10 (2). doi:10.14763/2021.2.1561. ISSN 2197-6775.

4.4. M. O'Dair, Blockchain: The Internetof Value, Springer International Publishing, Cham, 2019, pp. 15–30.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний засіб повинен виконувати процес автентифікації користувачів у системі.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Pentium 1500 МГц і подібні до них;

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3.

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	23.03.2024	02.04.2024	
2	Розробка захищеного веб-ресурсу	03.04.2024	12.04.2024	
3	Проектування веб-ресурсу	13.04.2024	25.04.2024	
4	Написання БДР на основі визначеної теми	26.04.2024	10.05.2024	
5	Тестування розробленого веб-ресурсу	11.05.2024	23.05.2024	
6	Попередній захист БДР	24.05.2024	02.06.2024	
7	Виправлення роботи	03.06.2024	11.06.2024	
8	Захист БДР	12.06.2024	17.06.2024	

10. Порядок контролю та прийому

10.1 До приймання бакалаврської дипломної роботи надається:

- ПЗ до бакалаврської дипломної роботи;
- демонстрація результату бакалаврської дипломної роботи;
- презентація;
- відзив керівника роботи;
- відзив рецензента

Технічне завдання до виконання прийняв _____ Стадник Р. Ю.

Додаток Б. Лістинг додатку

```
import threading
from flask import Flask, render_template, flash, redirect, url_for, session, request
from passlib.hash import sha256_crypt
from functools import wraps
from sqlhelpers import *
from forms import *
import time
app = Flask(__name__)
conn = sqlite3.connect('crypto.db', check_same_thread=False)
conn.row_factory = sqlite3.Row
_thread_local = threading.local()
def get_db():
    if not hasattr(_thread_local, 'conn'):
        _thread_local.conn = sqlite3.connect('crypto.db')
    return _thread_local.conn
def is_logged_in(f):
    @wraps(f)
    def wrap(*args, **kwargs):
        if 'logged_in' in session:
            return f(*args, **kwargs)
        else:
            flash("Unauthorized, please login.", "danger")
            return redirect(url_for('login'))
    return wrap
def log_in_user(username):
    users = Table("users", "name", "email", "username", "password")
    user = users.getone("username", username)
    session['logged_in'] = True
    session['username'] = username
    session['name'] = user['name']
    session['email'] = user['email']
@app.route("/register", methods = ['GET', 'POST'])
def register():
    form = RegisterForm(request.form)
    users = Table("users", "name", "email", "username", "password")
    if request.method == 'POST' and form.validate():
        #collect form data
        username = form.username.data
        email = form.email.data
        name = form.name.data
        if isnewuser(username):
            #add the user to mysql and log them in
            password = sha256_crypt.encrypt(form.password.data)
            users.insert(name,email,username,password)
            log_in_user(username)
            return redirect(url_for('dashboard'))
```

```
        else:
            flash('User already exists', 'danger')
            return redirect(url_for('register'))
    return render_template('register.html', form=form)
@app.route("/login", methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        candidate = request.form['password']
        users = Table("users", "name", "email", "username", "password")
        user = users.getone("username", username)
        # If the user cannot be found, the user does not exist
        if user is None:
            flash("Username is not found", 'danger')
            return redirect(url_for('login'))
        else:
            accPass = user.get('password')
            if sha256_crypt.verify(candidate, accPass):
                # Log in the user and redirect to Dashboard page
                log_in_user(username)
                flash('You are now logged in.', 'success')
                return redirect(url_for('dashboard'))
            else:
                # If the passwords do not match
                flash("Invalid password", 'danger')
                return redirect(url_for('login'))
    return render_template('login.html')
@app.route("/transaction", methods = ['GET', 'POST'])
@is_logged_in
def transaction():
    form = SendMoneyForm(request.form)
    balance = get_balance(session.get('username'))
    #if form is submitted
    if request.method == 'POST':
        try:
            #attempt to execute the transaction
            send_money(session.get('username'), form.username.data, form.amount.data)
            flash("Money Sent!", "success")
        except Exception as e:
            flash(str(e), 'danger')
        return redirect(url_for('transaction'))
    return render_template('transaction.html', balance=balance, form=form, page='transaction')
@app.route("/buy", methods = ['GET', 'POST'])
@is_logged_in
def buy():
    form = BuyForm(request.form)
    balance = get_balance(session.get('username'))
    if request.method == 'POST':
```

```
        #attempt to buy amount
        try:
            send_money("BANK", session.get('username'), form.amount.data)
            flash("Purchase Successful!", "success")
        except Exception as e:
            flash(str(e), 'danger')
        return redirect(url_for('dashboard'))
    return render_template('buy.html', balance=balance, form=form, page='buy')
@app.route("/logout")
@is_logged_in
def logout():
    session.clear()
    flash("Logout success", "success")
    return redirect(url_for('login'))
@app.route("/dashboard")
@is_logged_in
def dashboard():
    balance = get_balance(session.get('username'))
    blockchain = get_blockchain().chain
    ct = time.strftime("%I:%M %p")
    return render_template('dashboard.html', balance=balance, session=session, ct=ct,
blockchain=blockchain, page='dashboard')
@app.route("/")
@app.route("/index")
def index():
    return render_template('index.html')
#Run app
if __name__ == '__main__':
    app.secret_key = '12345'
    app.run(debug = True)
from hashlib import sha256
def updatehash(*args):
    hashing_text = ""; h = sha256()
    #loop through each argument and hash
    for arg in args:
        hashing_text += str(arg)
    h.update(hashing_text.encode('utf-8'))
    return h.hexdigest()
class Block():
    def __init__(self,number=0, previous_hash="0"*64, data=None, nonce=0):
        self.data = data
        self.number = number
        self.previous_hash = previous_hash
        self.nonce = nonce
    def hash(self):
        return updatehash(
            self.number,
            self.previous_hash,
```

```
            self.data,
            self.nonce
        )
    def __str__(self):
        return str("Block#: %s\nHash: %s\nPrevious: %s\nData: %s\nNonce: %s\n" %(
            self.number,
            self.hash(),
            self.previous_hash,
            self.data,
            self.nonce
            )
        )
class Blockchain():
    difficulty = 4
    def __init__(self):
        self.chain = []
    def add(self, block):
        self.chain.append(block)
    def remove(self, block):
        self.chain.remove(block)
    def mine(self, block):
        try: block.previous_hash = self.chain[-1].hash()
        except IndexError: pass
        while True:
            if block.hash()[:self.difficulty] == "0" * self.difficulty:
                self.add(block); break
            else:
                block.nonce += 1
    def isValid(self):
        for i in range(1,len(self.chain)):
            _previous = self.chain[i].previous_hash
            _current = self.chain[i-1].hash()
            #compare the previous hash to the actual hash of the previous block
            if _previous != _current or _current[:self.difficulty] != "0"*self.difficulty:
                return False
        return True
def main():
    blockchain = Blockchain()
    database = ["hello", "goodbye", "test", "DATA here"]
    num = 0
    for data in database:
        num += 1
        blockchain.mine(Block(num, data=data))
    for block in blockchain.chain:
        print(block)
    print(blockchain.isValid())
    blockchain.chain[2].data = "NEW DATA"
    blockchain.mine(blockchain.chain[2])
```

```
    print(blockchain.isValid())
if __name__ == '__main__':
    main()
from wtforms import Form, StringField, DecimalField, IntegerField, TextAreaField, PasswordField,
validators
class RegisterForm(Form):
    name = StringField('Full Name', [validators.Length(min=1,max=50)])
    username = StringField('Username', [validators.Length(min=4,max=25)])
    email = StringField('Email', [validators.Length(min=6,max=50)])
    password = PasswordField('Password', [validators.DataRequired(), validators.EqualTo('confirm',
message='Passwords do not match')])
    confirm = PasswordField('Confirm Password')
class SendMoneyForm(Form):
    username = StringField('Username', [validators.Length(min=4,max=25)])
    amount = StringField('Amount', [validators.Length(min=1,max=50)])
class BuyForm(Form):
    amount = StringField('Amount', [validators.Length(min=1,max=50)])
import sqlite3
from blockchain import Blockchain, Block
conn = sqlite3.connect('crypto.db', check_same_thread=False)
conn.row_factory = sqlite3.Row
class InvalidTransactionException(Exception): pass
class InsufficientFundsException(Exception): pass
class Table():
    def __init__(self, table_name, *args):
        self.table = table_name
        self.columns = ",".join(args)
        self.columnsList = args
        if isnewtable(table_name):
            create_data = ",".join([f"{column} TEXT" for column in self.columnsList])
            cur = conn.cursor()
            cur.execute(f"CREATE TABLE {self.table}({create_data})")
            conn.commit()
            cur.close()
    def getall(self):
        cur = conn.cursor()
        result = cur.execute(f"SELECT * FROM {self.table}")
        data = result.fetchall()
        cur.close()
        return data
    def getone(self, search, value):
        data = {}
        cur = conn.cursor()
        result = cur.execute(f"SELECT * FROM {self.table} WHERE {search} = ?", (value,))
        data = result.fetchone() if result else None
        cur.close()
        return data
    def deleteone(self, search, value):
```

```
        cur = conn.cursor()
        cur.execute(f"DELETE FROM {self.table} WHERE {search} = ?", (value,))
        conn.commit()
        cur.close()
    def deleteall(self):
        self.drop()
        self.__init__(self.table, *self.columnsList)
    def drop(self):
        cur = conn.cursor()
        cur.execute(f"DROP TABLE {self.table}")
        cur.close()
    def insert(self, *args):
        data = ",".join(["?" for _ in args])
        cur = conn.cursor()
        cur.execute(f"INSERT INTO {self.table}({self.columns}) VALUES({data})", args)
        conn.commit()
        cur.close()
def sql_raw(execution):
    cur = conn.cursor()
    cur.execute(execution)
    conn.commit()
    cur.close()
def isnewtable(tableName):
    cur = conn.cursor()
    try:
        cur.execute(f"SELECT * FROM {tableName}")
        cur.close()
        return False
    except sqlite3.OperationalError:
        return True
def isnewuser(username):
    # Access the users table and get all values from column "username"
    users = Table("users", "name", "email", "username", "password")
    data = users.getall()
    usernames = [user['username'] for user in data]
    return False if username in usernames else True
def send_money(sender, recipient, amount):
    try: amount = float(amount)
    except ValueError:
        raise InvalidTransactionException("Invalid Transaction.")
    if amount > get_balance(sender) and sender != "BANK":
        raise InsufficientFundsException("Insufficient Funds.")
    elif sender == recipient or amount <= 0.00:
        raise InvalidTransactionException("Invalid Transaction.")
    elif isnewuser(recipient):
        raise InvalidTransactionException("User Does Not Exist.")
    blockchain = get_blockchain()
    number = len(blockchain.chain) + 1
```

```
    data = f"{sender}-->{recipient}-->{float(amount)}"
    blockchain.mine(Block(number, data=data))
    sync_blockchain(blockchain)
def get_balance(username):
    balance = 0.00
    blockchain = get_blockchain()
    # Loop through the blockchain and update balance
    for block in blockchain.chain:
        data = block.data.split("-->")
        if username == data[0]:
            balance -= float(data[2])
        elif username == data[1]:
            balance += float(data[2])
    return balance
def get_blockchain():
    blockchain = Blockchain()
    blockchain_sql = Table("blockchain", "number", "hash", "previous", "data", "nonce")
    for b in blockchain_sql.getall():
        blockchain.add(Block(int(b['number']), b['previous'], b['data'], int(b['nonce'])))
    return blockchain
def sync_blockchain(blockchain):
    blockchain_sql = Table("blockchain", "number", "hash", "previous", "data", "nonce")
    blockchain_sql.deleteall()
    for block in blockchain.chain:
        blockchain_sql.insert(block.number, block.hash(), block.previous_hash, block.data, block.nonce)
(function() {
  const doc = document.documentElement;
  doc.classList.remove("no-js");
  doc.classList.add("js");
  // Reveal animations
  if (document.body.classList.contains("has-animations")) {
    /* global ScrollReveal */
    const sr = (window.sr = ScrollReveal());
    sr.reveal(".hero-title, .hero-paragraph, .hero-cta", {
      duration: 1000,
      distance: "40px",
      easing: "cubic-bezier(0.5, -0.01, 0, 1.005)",
      origin: "bottom",
      interval: 150
    });
    sr.reveal(".feature, .pricing-table", {
      duration: 600,
      distance: "40px",
      easing: "cubic-bezier(0.5, -0.01, 0, 1.005)",
      interval: 100,
      origin: "bottom",
      viewFactor: 0.5
    });
```

```
    sr.reveal(".feature-extended-image", {
      duration: 600,
      scale: 0.9,
      easing: "cubic-bezier(0.5, -0.01, 0, 1.005)",
      viewFactor: 0.5
    });
  }
})();
(function($) {
  "use strict"; // Start of use strict
  // Toggle the side navigation
  $("#sidebarToggle").on("click", function(e) {
    e.preventDefault();
    $("body").toggleClass("sidebar-toggled");
    $(".sidebar").toggleClass("toggled");
  });
  // Prevent the content wrapper from scrolling when the fixed side navigation hovered over
  $("body.fixed-nav .sidebar").on("mousewheel DOMMouseScroll wheel", function(
    e
  ) {
    if ($(window).width() > 768) {
      var e0 = e.originalEvent,
        delta = e0.wheelDelta || -e0.detail;
      this.scrollTop += (delta < 0 ? 1 : -1) * 30;
      e.preventDefault();
    }
  });
  // Scroll to top button appear
  $(document).on("scroll", function() {
    var scrollDistance = $(this).scrollTop();
    if (scrollDistance > 100) {
      $(".scroll-to-top").fadeIn();
    } else {
      $(".scroll-to-top").fadeOut();
    }
  });
  // Smooth scrolling using jQuery easing
  $(document).on("click", "a.scroll-to-top", function(event) {
    var $anchor = $(this);
    $("html, body")
      .stop()
      .animate(
        {
          scrollTop: $($anchor.attr("href")).offset().top
        },
        1000,
        "easeInOutExpo"
```

Додаток В. Ілюстративний матеріал

Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайнторгівлі за допомогою блокчейн-технології

Виконав студент групи 1КІТ С-20б Стадник Р. Ю.

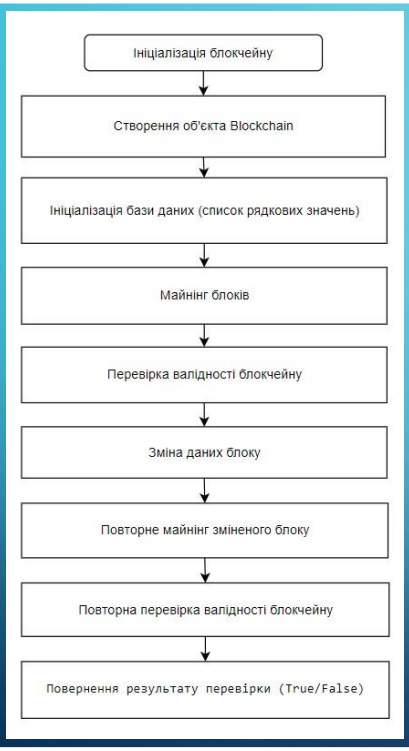
Науковий керівник: к.т.н., доцент кафедри МБІС Грицак А. В.

МЕТА РОБОТИ

Мета полягає у створенні зручної платформи для онлайн трейдингу, що основана на технології блокчейн, яка буде давати змогу користувачам обмінюватись криптовалютою, можливість покупки валюти, а також статистичну інформацію для спостереження за нею

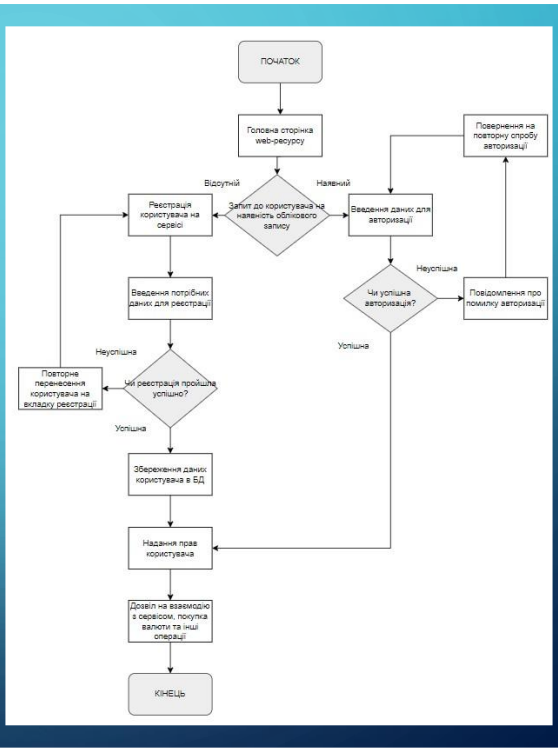
ПЕРЕВАГИ ВИКОРИСТАНОЇ ТЕХНОЛОГІЇ БЛОКЧЕЙН

Блокчейн складається з цифрової мережі розподілених по всьому світу комп'ютерів, так званих вузлів, які записують дані та історію транзакцій, що і дає йому можливість забезпечувати високий захист, а також важливим аспектом є хешування інформації, що передбачає перетворення вхідних даних у вихід фіксованого розміру за допомогою конкретного алгоритму



АЛГОРИТМ РОБОТИ СТВОРЕНОГО БЛОКЧЕЙНУ

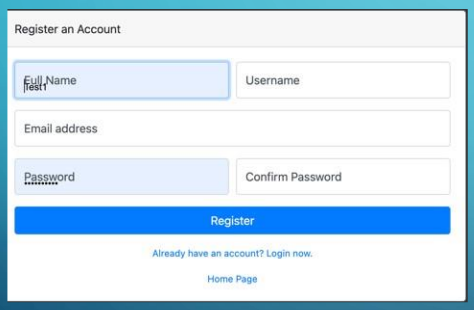
СХЕМА ВЗАЄМОДІЇ КОРИСТУВАЧА З СЕРВІСОМ

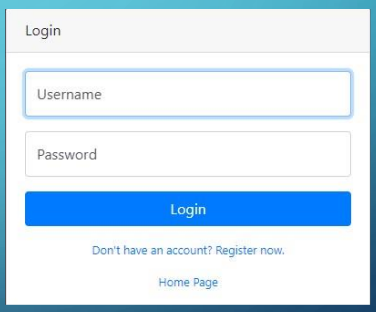


ТЕХНОЛОГІЇ, ЩО БУЛИ ВИКОРИСТАНІ У РОЗРОБЦІ

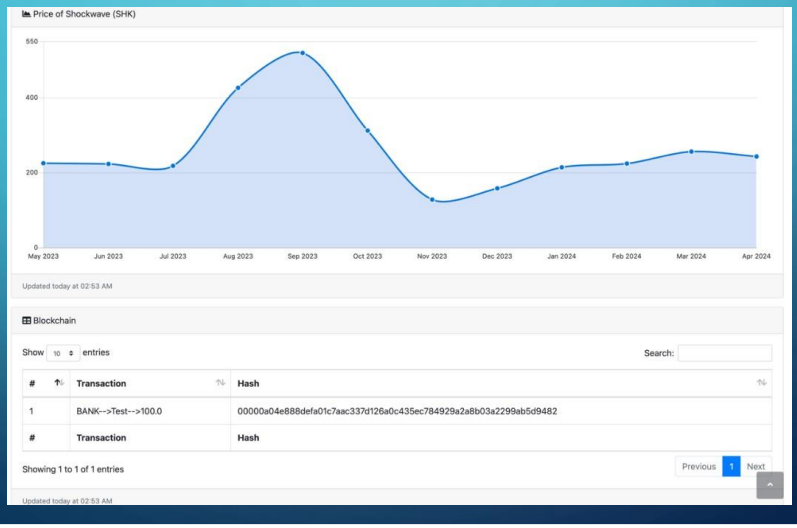
- Java Script
- Python
- MySQL
- Flask-MySQLdb
- HTML
- SASS
- Bootstrap

ПРОЦЕДУРИ РЕЄСТРАЦІЇ ТА ЛОГІНІЗАЦІЇ КОРИСТУВАЧА

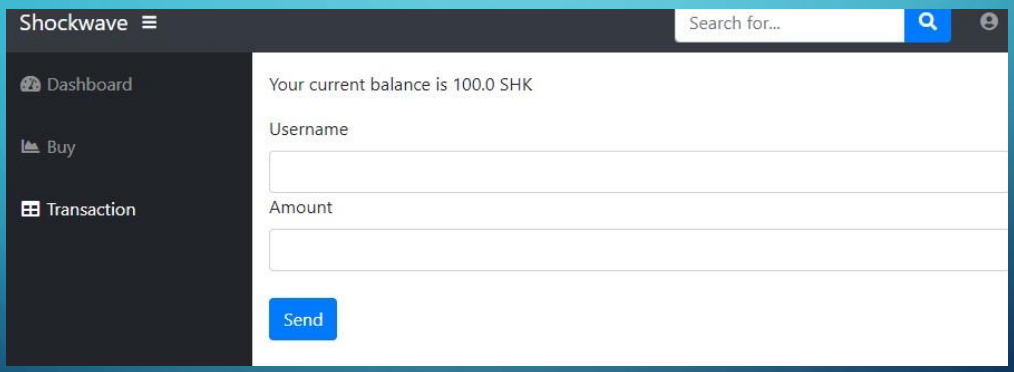




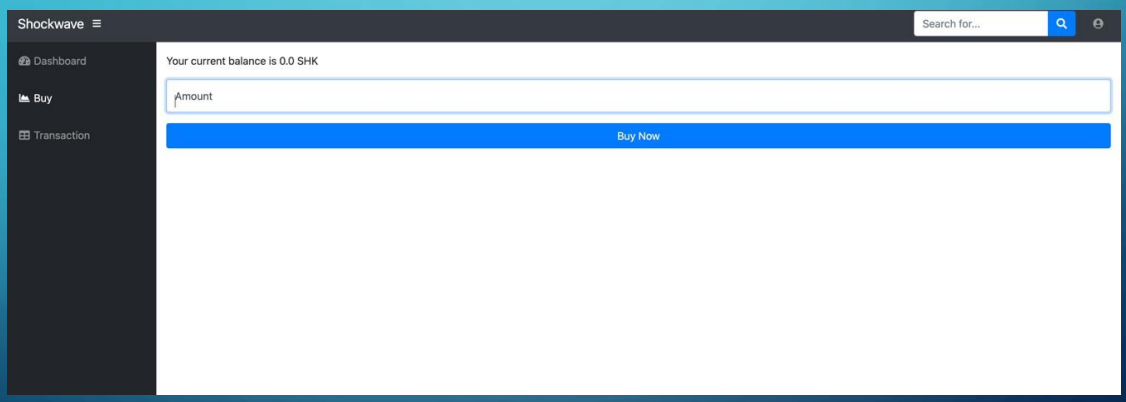
СТАТИСТИЧНІ ДАНІ ПО КРИПТОВАЛЮТІ



ПЕРЕДАЧА ВАЛЮТИ ІНШИМ КОРИСТУВАЧАМ



ВІКНО ПОКУПКИ КРИПТОВАЛЮТИ



ВИСНОВКИ ДО РОБОТИ

Під час виконання роботи було досягнуто наступні цілі:

1. Дослідження та порівняння існуючих аналогів.
2. Розробка алгоритмів системи, схем взаємодії з програмою та бази даних.
3. Реалізація веб-додатку для онлайн трейдингу.
4. Тестування розробленої платформи.

В кінцевому результаті було отримано захищений корпоративний веб -ресурс для онлайн трейдингу з використанням блокчейн технології, що надає користувачам можливість користуватися криптовалютою, тобто купляти, передавати іншим користувачам, а також спостерігати зміни в блокчейні.

УСІМ ДЯКУЮ ЗА УВАГУ

Додаток Г. Протокол перевірки на антиплагіат

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ
ЗАПОЗИЧЕНЬ

Назва роботи: Розробка захищеного корпоративного web-ресурсу для надання послуг з онлайн торгівлі за допомогою блокчейн-технології

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 96% Схожість 4%

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Стадник Р.Ю.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Грицак А.В.
(підпис) (прізвище, ініціали)