Хмельницький національний університет
Факультет програмування
та комп'ютерних і телекомунікаційних систем
Кафедра інженерії програмного забезпечення

ДИПЛОМНА РОБОТА

Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum

Рівень вищої освіти Другий (магістерський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДРІПЗ.150181.01.08.ПЗ

Виконав студент 2 курсу група ІПЗм-19-1 _____ М. Л. Хорошун
Підпис Ініціали, прізвище

Керівник канд. техн. наук, доцент _____ Г. І. Радельчук
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. техн. наук, доцент _____ Г. І. Радельчук
Підпис Ініціали, прізвище

До захисту допускаю:
Завідувач кафедри інженерії
програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

7 грудня 2020 р.

Хмельницький 2020

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Програмування та комп'ютерних і телекомунікаційних систем
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Другий (магістерський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
Л. П. Бедратюк
02 09 2020 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Хорошуну Михайлу Леонтійовичу
Прізвище, ім'я, по батькові студента

1. Тема проєкту (роботи Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum

Керівник проєкту (роботи) Радельчук Галина Іванівна
кандидат технічних наук, доцент

Затверджена наказом ректора університету від 01.09.2020 р. № 118

2. Строк подання студентом проєкту (роботи) на кафедру 01.12.2020 р.

3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1 Дослідження предметної області та постановка задачі
2 Концепції, моделі та методи вирішення задачі
3 Алгоритми та технології вирішення задачі
4 Реалізація та тестування програмної системи

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди)

6. Консультанти розділів дипломного проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 01 вересня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1 Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження; визначення структури дипломної роботи	01.09 – 07.09.2020	
2 Робота над розділом 1 дипломної роботи – вивчення літературних джерел; аналіз відомих моделей, методів та засобів за темою роботи; висновки до розділу та постановка задачі	08.09 – 25.09.2020	
3 Робота над розділом 2 дипломної роботи – розробка моделей та методів вирішення поставленої задачі; висновки до розділу	26.09 – 10.10.2020	
4 Робота над науковими статтями	11.10 – 20.10.2020	
5 Робота над розділом 3 дипломної роботи – розробка алгоритмів та технологій, проектування ПЗ для вирішення поставленої задачі; висновки до розділу	11.10 – 26.10.2020	
6 Робота над розділом 4 дипломної роботи – програмна реалізація спроектованих рішень, результати експериментів, їх аналіз; висновки до розділу	27.10 – 15.11.2020	
7 Узгодження постановки задачі, отриманих результатів та висновків; написання вступу, загальних висновків, оформлення джерел посилання та додатків; оформлення пояснювальної записки та графічних матеріалів згідно вимог стандартів	16.11 – 30.11.2020	
8 Попередній захист дипломної роботит	Листопад (згідно графіка)	
9. Перевірка роботи на наявність плагіату; нормоконтроль; брошурування пояснювальної записки; підготовка супровідних документів	01.12 – 04.12.2020	
10 Підготовка до захисту дипломної роботи	05.12 – 08.12.2020	

Студент _____ М. Л. Хорошун
Підпис Ініціали, прізвище

Керівник проєкту (роботи) _____ Г. І. Радельчук
Підпис Ініціали, прізвищ

РЕФЕРАТ

Тема дипломної роботи: «Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum».

Автор роботи: Хорошун Михайло Леонтійович.

Керівник роботи: Радельчук Галина Іванівна.

Пояснювальна записка: 90 с., 18 рис., 9 табл., 6 дод., 19 джерел.

ЦИФРОВА ВАЛЮТА, БЛОКЧЕЙН, ETHEREUM, ДЕЦЕНТРАЛІЗОВАНА ПЛАТІЖНА СИСТЕМА, РОЗУМНИЙ КОНТРАКТ, SOLIDITY.

Об’єктом дослідження є процеси функціонування децентралізованої платіжної системи та програмно-технічна база, необхідна для забезпечення обігу цифрової валюти в її рамках.

Мета дослідження – розробка програмного комплексу для децентралізованої платіжної системи з обігу власної криптовалюти.

У роботі використані наступні методи дослідження та апаратура:

– спостереження, експеримент, абстрагування, аналіз та синтез, формалізація;

– сучасні інструментальні засоби проектування та програмування;

– персональний комп’ютер.

У дипломній роботі досліджено процедуру залучення криптовалютних інвестицій; визначено проблеми, наявні у галузі, та шляхи їх вирішення; сформовано основні вимоги до створюваної системи та функцій, які вона повинна виконувати.

У дослідженні удосконалено метод створення цифрових платіжних засобів шляхом проектування комплексного рішення, яке складається з клієнтської частини та системи розумних контрактів. Обгрунтовано доцільність проектування власної обмінної платформи поряд з інтеграцією зовнішніх криптовалютних бірж для можливості купівлі створеної криптовалюти за інші популярні цифрові валюти. Удосконалено методи опрацювання транзакцій, що дозволило оптимізувати пропускну здатність системи у порівнянні з існуючими рішеннями. Розглянуто алгоритми функціонування блокчейн-систем та обрано оптимальний алгоритм консенсусу.

Результатом дослідження є покращені методи проектування децентралізованих платіжних систем.

На основі визначених вимог розроблено функціональну модель платіжної системи та виконано її програмну реалізацію. Для розробки програмної системи використані наступні технології: блокчейн-платформа Ethereum; мови програмування Solidity, Javascript; фреймфорки NodeJs, Feathers, Truffle (для розробки), Mocha, Char, Solidity-Coverage (для тестування).

У підсумку проведено апробацію отриманих результатів та впровадження програмної системи. Завдяки поліпшеним характеристикам, порівняно з традиційними рішеннями, розроблена програмна система має високі конкурентні шанси на ринку. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у притоках інвестиційного капіталу у формі криптовалют.

04.12.2020р

ABSTRACT

Master’s thesis: «Decentralized Payment System with its Own Digital Currency and Cryptographic Protection Based on Ethereum Blockchain Platform».

Author: Khoroshun Mykhailo Leontiyovich.

Head of work: Galina Ivanovna Radelchuk.

Master’s thesis consists of: 90 p., 18 pc., 9 tb., 6 add., 19 srs.

DIGITAL CURRENCY, BLOCKCHAIN, ETHEREUM, DECENTRALIZED PAYMENT SYSTEM, SMART CONTRACT, SOLIDITY

The subject of the research are the processes of functioning of the decentralized payment system and the software necessary to ensure the circulation of digital currency within it.

The purpose of the study is to develop a software for a decentralized payment system for the circulation of its own cryptocurrency.

The following research methods and equipment were used in the study:

– observation, experiment, abstraction, analysis and synthesis, formalization;

– modern tools for design and programming;

– personal computer.

The study examined the procedure for attracting cryptocurrency investments; identified problems in the industry and ways to solve them; formed the basic requirements for the system and the functions it must perform.

The study explored ways to improve the method of creating digital means of payment by designing a comprehensive solution that consists of a client part and a system of smart contracts. The study substantiated the feasibility of designing its own exchange platform along with the integration of external cryptocurrency exchanges for the possibility of buying the created cryptocurrency for other popular digital currencies. During the research, the methods of transaction processing were improved, which allowed to optimize the system bandwidth in comparison with existing solutions, the algorithms of blockchain systems operation were considered and the optimal consensus algorithm was chosen. The result of the study is improved methods of designing decentralized payment systems.

Based on the defined requirements, a functional model of the payment system was developed along with its software implementation. The following technologies were used to develop the software system: Ethereum blockchain platform; programming languages: Solidity, Javascript; framewors: NodeJs, Feathers, Truffle (for development), Mocha, Char, Solidity-Coverage (for testing).

In the end, the obtained results were tested and the software system was implemented. Due to the improved characteristics compared to traditional solutions, the developed software system has a high competitive chance in the market. Therefore, it is recommended to integrate this system for companies that are interested in inflows of investment capital in the form of cryptocurrencies.

4.12.2020 р.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД	–	база даних
БС	–	банківська система
ВВ		варіант використання
ДПС	–	децентралізована платіжна система
ІК	–	інтерфейс користувача
МП	–	мова програмування
ПЗ	–	програмне забезпечення
ПС	–	програмна система
СКБД	–	система керування базами даних
API	–	Application Programming Interface
DA	–	Decentralized Applications
ECDSA	–	Elliptic Curve Digital Signature Algorithm
JSON-RPC	–	JavaScript Object Notation Remote Procedure Call
MVC	–	Model-View-Controller
PBFT	–	Practical Byzantine Fault Tolerance
PoS	–	Proof-of-Stake
PoW	–	Proof-of-Work
RPC	–	Remote Procedure Call
UML	–	Unified Modeling Language

ВСТУП

На сучасному етапі розвитку людства гроші втратили товарну сутність і сприймаються лише як розрахункова одиниця. Але еволюція грошей продовжується – паперові гроші перетворюються на цифрові, серед яких все більше виокремлюються ДПС як альтернатива банкам. ДПС дозволяють виключити банківську систему з процесу емісії грошей та проведення транзакцій і довірити це комп'ютерним алгоритмам. Такі системи не мають обмежень у формуванні обмінних курсів та здійсненні операцій, дозволяють виконувати вільне переміщення грошових коштів. Транзакції в таких системах не піддаються цензурі та є незворотніми.

Поява ДПС продемонструвала, що криптовалюти можуть бути ефективним інструментом інвестування. Випуск віртуальної цифрової валюти, яку потім можна використовувати як платіжний засіб усередині сервісу або компанії, виявився найпростішим та найпривабливішим способом як для залучення інвестицій, так і для інвестування. Таким чином з'явився попит на створення ДПС з власними цифровими валютами. Зазвичай, криптовалюта у таких системах виступає у ролі внутрішньої валюти додатку. Однак, методи реалізації подібних систем різняться за показниками ефективності, безпеки коштів та цінністю валюти на глобальному ринку.

До прикладу, безумовною перевагою криптовалют є можливість здійснення прямих платежів між користувачами, відсутність національних кордонів для здійснення переказів та зниження операційних витрат у порівняннні з традиційними БС. Але, окрім внутрішніх витрат за перекази коштів між користувачами мережі для здійснення оплати у криптовалюті, користувачу, який не має її у своєму розпорядженні, необхідно спершу обміняти наявні у нього гроші на криптовалюту (що потребує використання онлайн-бірж або обмінників). Така конвертація валют містить додаткові комісійні витрати. При здійсненні платежів із використанням криптовалют транзакційні витрати всередині ДПС можуть бути досить низькими і, враховуючи можливості транскордонного переміщення коштів, привабливішими у порівнянні із банківськими платежами (але якщо користувачу необхідно здійснювати обмін або купувати криптовалюту для здійснення такого платежу, то додаткові комісійні витрати можуть перевищувати аналогічні у БС).

Також однією з головних проблем для ДПС залишається масштабованість. Наприклад, у рамках протоколу Bitcoin блок транзакцій обмежений розміром в 1 Мб і швидкість їх обробки становить приблизно сім операцій за секунду, в той час як Visa обробляє у середньому 2000 операцій за секунду. Розмір блоку впливає на кількість транзакцій, які можна додати у блок. Протокол Bitcoin передбачає, що блок формується у середньому 10 хвилин, і при збільшення активності у мережі збільшуються як комісійні, що пропонуються відправниками, так і час підтвердження окремої транзакцій вузлами мережі.

Ще одним важливим моментом при здійсненні платежів є спосіб підтвердження транзакцій. При використанні централізованих БС банк є посередником і гарантом переміщення коштів між рахунками клієнтів. У ДПС визначення, чи є транзакція вірною, відбувається на основі консенсусу учасників такої системи. Тобто її підтвердження здійснюється «більшістю голосів». Понад 90% існуючих систем використовують алгоритм консенсусу PoW, суть якого зводиться до того, що десятки тисяч комп'ютерів витрачають власні обчислювальні ресурси на виконання протоколу консенсусу і при цьому лише один з них наприкінці отримує можливість створити блок. У результаті це призводить до великих енергозатрат, що є проблемою.

Таким чином, проаналізувавши наукові дослідження та публікації, можна зробити висновок, що основними проблемами, пов'язаними з існуючими ДПС є: низька швидкість проходження транзакцій; необхідність залучення третьої сторони (криптовалютних бірж) для купівлі/обміну власної криптовалюти; функціонування системи на алгоритмі консенсусу PoW, що вимагає значних енергозатрат. Однак, всі ці задачі можна вирішити, покращивши алгоритми та моделі традиційної системи.

Отже, актуальність теми роботи полягає у необхідності розробки повнофункціональної ДПС, яка б дозволяла здійснювати перекази цифрової валюти, відповідала високим стандартам безпеки та була незалежною від банківських регуляторів. При цьому система має вирішувати проблеми традиційних рішень.

Мета дослідження – розробка програмного комплексу для децентралізованої платіжної системи з обігу власної криптовалюти.

Завдання дослідження, які необхідно вирішити для досягнення мети:

– проаналізувати специфіку функціонування ДПС;

– дослідити процедури залучення криптовалютних інвестицій та обґрунтувати необхідність розробки системи для обігу криптовалют;

– провести аналіз існуючих моделей та методів у галузі криптовалютних інвестицій, виділити невирішені проблеми;

– визначити основні вимоги до ДПС та функції, які вона має виконувати;

– удосконалити моделі та методи організації процесу функціонування ДПС, які б вирішували наявні проблеми у галузі криптовалютних інвестицій;

– виконати проектування ПС на основі розроблених методів та алгоритмів;

– виконати програмну реалізацію прийнятих рішень;

– провести тестування та практичну апробацію отриманих результатів;

– проаналізувати отримані результати та сформувати рекомендації щодо доцільності впровадження результатів дослідження.

Об’єкт дослідження – процеси функціонування ДПС та програмно-технічна база, необхідна для забезпечення обігу цифрової валюти у рамках ДПС.

Предмет дослідження – моделі, методи та механізми створення безпечної, прозорої та ефективної ДПС, яка б забезпечувала процес обігу цифрової валюти.

Методи, які були використані у роботі для досягнення мети, є наступними.

Емпіричні методи

Спостереження. Темою роботи є побудова моделей та механізмів, пов’язаних із технологією блокчейн, але для того, щоб виділити корисні ознаки, які мають бути імплементовані у розроблюваних рішеннях, слід провести спостереження над існуючими аналогами, визначити властивості та зв’язки між ними.

Експеримент. На етапі дослідження існуючих аналогів слід відтворити певні умови, які потрібні для аналізу імплементованих алогоритмів. Пізніше цей же метод використовується для аналізу ефективності результативної моделі, яка розробляється та імплементується у ході роботи.

Теоретичні методи:

– абстрагування – один з важливих методів, який дозволяє відкинути несуттєві параметри; від абстрагування напряму залежить ефективність моделі;

– аналіз та синтез – декомпозиція моделі на прості складові, виявлення зв’язків між компонентнами; відповідно, і синтез цих структурних елементів у єдине ціле;

– формалізація – представлення моделі у вигляді програмного коду.

Наукова новизна отриманих результатів:

– удосконалено метод створення цифрових платіжних засобів шляхом розробки комплексного рішення, що складається з блокчейн-платформи та системи розумних контрактів;

– удосконалено метод виходу криптовалюти на відкритий ринок шляхом розробки власної криптовалютної електронної біржі;

– удосконалено метод знаходження консенсусу мережі, що дозволить збільшити її пропускну здатність та зменшити витрати електроенергії на її утримання;

– удосконалено метод розподілення винагороди за підтримання консенсусу мережі.

Практична цінність отриманих результатів полягає в успішній розробці моделей та механізмів забезпечення безпечного, прозорого та ефективного процесу створення власної криптовалюти. Завдяки поліпшеним характеристикам, порівняно з традиційними рішеннями, розроблена ПС має високі конкуретні шанси на ринку. Результати практичної апробації ПС підтверджують її працездатність та відповідність вимогам безпеки. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у притоках інвестиційного капіталу у формі криптовалют.

Достовірність та обґрунтованість отриманих результатів підтверджується використанням у процесі дослідження таких прийомів:

– перевірка теоретичних положень, нових рішень, ідей експериментальними дослідженнями за допомогою відомих процедур проектування та тестування;

– працездатність та функціональна придатність розробленої ПС;

– наявність наукової публікації у рецензованому виданні.

За темою дипломної роботи опубліковано дві наукові статті: одна стаття – у фаховому науковому виданні та одна стаття – у збірнику матеріалів Міжнародної науково-практичної конференції.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної області, останніх досліджень та джерел

Уже давно гроші стали основою обміну товарами і послугами між людьми. Переглянувши історію розвитку грошей, можна відстежити постійний процес трансформації як самих грошей, так і ставлення людей до них. Багато років у якості грошей використовували метали, які можна було не лише зберігати з найменшою втратою, але й ділити без втрат на потрібну кількість частин, які потім можуть бути легко переплавлені в один шматок. У ході історії еволюція грошей привела до появи паперових грошей, вартість який була забезпечена золотом, а згодом і кредитних грошей у вигляді національних валют, які вже не мали ніякого забезпечення, окрім суспільної гарантії в особі певної держави. Тобто традиційні гроші втратили товарну сутність і сьогодні вони сприймаються лише як розрахункова одиниця. Але еволюція грошей продовжується – паперові гроші перетворюються на цифрові, серед яких все більше виокремлюються децентралізовані платіжні системи як альтернатива традиційним грошам центральних банків.

Як підкреслюють науковці З. Васильченко та Д. Пасевич, сьогодні все більше набирає популярності тенденція до застосування безконтактних платежів [1]. Цифрові платіжні системи стають все більш популярними – вони є зручнішими для користувачів у порівнянні з готівкою та забезпечують кращий контроль за грошовими потоками. За даними Національного банку України за період 2013-2018 років частка безготівкових операцій в Україні зросла з 17,4% до 45,1% [2]. Однак, варто зауважити, що використання цифрових грошей базується на довірі до фінансової установи, яка не завжди гарантує 100% покриття за рахунком. І при настанні несприятливих обставин існує ймовірність втратити довірені кошти (іншими словами, клієнт банку, який зберігає гроші на рахунку, не є їх повноцінним власником).

Як альтернатива, на сьогодні активно розвиваються децентралізовані платіжні системи, що дозволяють взагалі виключити банківську систему з процесу емісії грошей та проведення транзакцій і довірити це комп’ютерним алгоритмам. Крипто-

валюта стає інноваційним фінансовим інструментом як децентралізований цифровий вимір вартості, що має криптографічний захист обліку [3].

Децентралізовані системи не мають обмежень у формуванні обмінних курсів та здійсненні операцій, дозволяють виконувати вільне переміщення грошових коштів без обмежень національними кордонами. Транзакції в таких системах не піддаються цензурі та є незворотніми. Порівняльна характеристика централізованих та децентралізованих платіжних систем подана у таблиці 1.1.

Таблиця 1.1 – Характеристика централізованих та децентралізованих систем

Централізована платіжна система	Децентралізована платіжна система
Обробка транзакцій здійснюється на виділеному сервері або дата центрі	Мережа центрів, кожен з яких має однаковий доступ до виконання транзакцій
Для зберігання даних використовується база даних	Для зберігання даних використовується децентралізована база даних
Для виконання транзакцій необхідний номер рахунку та пароль користувача	Для виконання транзакцій необхідний електронний цифровий підпис
Курс валюти регулюється державою	Курс валюти регулюється запитом
Емісія валюти регулюється державою	Емісія валюти регулюється програмним алгоритмом
Виконані транзакції можуть бути відкликані	Транзакції є незворотніми

Децентралізована платіжна система – це цифрова пірингова платіжна система, яка використовує розрахункову одиницю (криптовалюту) для обліку операцій. Така комп’ютерна мережа заснована на рівноправності учасників (тобто відсутні виділені сервери), а кожний вузол одночасно виконує функцію як клієнта, так і сервера. Подібна організація дозволяє зберігати працездатність мережі при будь-якій кількості і будь-якому поєднанні доступних вузлів. Функціонування та захист системи забезпечуються використанням криптографічних методів. При цьому вся інформація про транзакції між адресами системи доступна у відкритому вигляді [4].

Поява ДПС продемонструвала, що криптовалюти можуть бути ефективним інструментом інвестування. Випуск віртуальної цифрової валюти, яку потім можна використовувати як платіжний засіб усередині сервісу або компанії, виявився най-

простішим і найпривабливішим способом як для залучення інвестицій, так і для інвестування. Таким чином, з'явився попит на створення ДПС з власними криптовалютами, «заточені» під сервісні потреби бізнесу. Зазвичай, криптовалюта у таких системах виступає у ролі внутрішньої валюти додатку. Однак, методи та засоби реалізації подібних систем різняться за показниками ефективності, безпеки коштів та цінністю валюти на глобальному ринку [5]. Тому є сенс розглянути їх детальніше.

1.2 Аналіз існуючих методів та засобів у галузі криптовалютних інвестицій

Загальні методи реалізації нового цифрового активу можна розділити на три групи [6]:

– створення власної реалізації блокчейн-системи;

– розміщення активу на існуючих платформах;

– доопрацювання відкритих реалізацій блокчейн-протоколів.

Найпростішим та найпопулярнішим способом створення своєї цифрової валюти є її розміщення на вже існуючих ресурсах – блокчейн-майданчиках. Для кожної подібної мережі існує свій набір правил використання та взаємозамінності цифрового активу. Цифрова валюта, розроблена подібним способом, функціонує на технології готової «батьківської» платіжної системи, а сама прозорість логіки функціонування досягається шляхом написанням спеціальних програм – розумних контрактів, які здійснюють контроль над інвестиціями.

Розумний контракт (Smart Contract) – це комп'ютерний протокол, призначений для цифрового укладення, зміни, виконання та розірвання угод. Розумні контракти дозволяють здійснювати ці операції надійно та без сторонніх осіб. Взаємодія з контрактом є відстежуваною та незворотньою. Метою розумних контрактів є забезпечення вищого рівня безпеки, порівняно з традиційними договірними правами і зменшення транзакційних витрат, пов'язаних з функціонуванням договору [7].

На сучасному етапі найпопулярнішими блокчейн-майданчиками є платформи Ethereum, NEO, NEM, Ripple та Waves. Особливо популярною є мережа Ethereum.

Ethereum – загальнодоступна розподілена обчислювальна платформа на базі блокчейну. Ця платформа реалізована як єдина децентралізована віртуальна система, що дозволяє виконувати розумні контракти. Ethereum здатна виконувати код на семи мовах програмування, повних за Тьюрингом. Розробники використовують ці мови для створення та публікації розумних контрактів, які працюють всередині Ethereum. Програми, побудовані на платформі Ethereum, зазвичай називають децентралізованими додатками (DA), оскільки вони базуються на децентралізованій віртуальній машині Ethereum та її розумних контрактах [8]. Це означає, що будь-яка інформація про транзакції розділяється та відправляється на різні пристрої зберігання, які можуть знаходитись один від одного у сотнях, а то й тисячах кілометрів. Таким чином, виключається можливість шахрайства чи доступу до даних для сторонніх осіб [9].

Стандартизація криптовалютних розумних контрактів дозволила створити на платформі Ethereum широкий спектр нових цифрових валют, зробивши її найпопулярнішою розподіленою платформою для збору коштів та залучення інвестицій. До травня 2018 року, згідно з даними порталу coinmarketcap.com, в обігу перебуває понад 1600 криптовалют.

Серед переваг створення цифрової валюти на Ethereum чи подібних блокчейн-майданчиках є простота реалізації та зручність для інвесторів, які вже є учасниками цієї платіжної системи.

До недоліків подібних систем відносяться:

– прив’язка до валюти «батьківської» системи (для того, щоб зробити інвестицію в таку криптовалюту, потрібно попередньо придбати криптовалюту «батьківської» платформи);

– неможливість оптимізовувати пропускну здатність (транзакції всіх підсистем відбуваються в рамках «батьківської» платформи, тому навантаження на одну з підсистем призводить до перенавантаження всієї системи);

– виконання усіх транзакцій супроводжується комісійними відрахуваннями у валюті «батьківської» системи.

– розроблена цифрова валюта існує, поки існує «батьківська» система.

Перераховані проблеми можна було б вирішити шляхом розробки власної криптовалюти з нуля, але очевидною проблемою такого підходу буде висока складність розробки, тестування та відлагодження, що може тягнутись роками і потребувати значних фінансових вливань та великої команди розробки.

Таким чином, найгнучкішим способом реалізації платіжної системи з власною криптовалютою є доопрацювання реалізації готових блокчейн-протоколів. Компанія, яка емітує монету таким способом, має інструменти для гнучкої персоналізації системи, її пропускної здатності, персоналізації деталей функціонування протоколу тощо.

Для того, щоб монета мала популярність серед криптоінвесторів, вона повинна бути розроблена разом з комплексом додаткових сервісів та інструментів і відповідати вимогам безпеки, надійності та продуктивності. Тому існуючі підходи для реалізації платіжних систем розділяють за критеріями, що включають [10]:

– приватний або публічний блокчейн;

– алгоритм консенсусу.

У публічного і приватного блокчейнів є багато спільного: вони обидва є децентралізованими одноранговими мережами, де кожний учасник підтримує репліку загального реєстру та має право додавання транзакцій з цифровим підписом. Обидва блокчейни підтримують алгоритми синхронізації реплік через протоколи консенсусу, а також надають певні гарантії незмінності реєстру (навіть коли деякі учасники виявились дефектними або зловмисними).

Відмінність між публічним і приватним блокчейном пов’язана з тим, кому дозволено брати участь у мережі та виконувати узгоджений протокол. Публічна мережа є повністю відкритою, тобто будь-хто може стати її учасником. Приватний блокчейн – це мережа, якою керує одна організація і в її роботі можна приймати участь лише за запрошенням цієї організації. В учасників такої мережі мають бути спеціальні дозволи на читання, запис чи перевірку блокчейна.

І публічний, і приватний блокчейни можуть мати свою власну криптовалюту. У публічному блокчейні вона може використовуватися для оплати комісії за транзакції та стимулювати підтримку мережі. Монети цієї криптовалюти можуть бути використані безпосередньо у самому додатку. При цьому криптовалюта у

приватному блокчейні не матиме жодної реальної цінності, так як для її забезпечення монета повинна мати зовнішню цінність за межами блокчейну. Таким чином, створення криптовалюти на основі приватного блокчейну не має під собою обґрунтованих причин.

У процесі застосування технології блокчейн виникає багато проблем та питань, серед яких основним є: як розробити відповідний протокол консенсусу. Консенсус блокчейна полягає в тому, що всі вузли підтримують однаковий розподілений реєстр. У традиційній архітектурі ПЗ консенсус навряд чи є проблемою через існування центрального сервера, отже, інші вузли потрібно лише узгодити з сервером. Однак, у розподіленій мережі, такій як блокчейн, кожен вузол є і хостом, і сервером, і йому потрібно обмінюватися інформацією з іншими вузлами, щоб досягти консенсусу. Інколи деякі вузли будуть працювати в режимі офлайн. Крім того, можуть з'явитись деякі шкідливі вузли, які будуть негативно впливати на процес консенсусу і, навіть, можуть зашкодити йому. Тому консенсус-протокол має не допустити виникнення цих ситуацій та мінімізувати шкоду, щоб не вплинути на кінцевий результат консенсусу.

Безумовною перевагою криптовалют є можливість здійснення прямих платежів між користувачами, відсутність національних кордонів для здійснення переказів та зниження операційних витрат у порівнянні з традиційними банківськими системами. Але, окрім внутрішніх витрат за перекази коштів між користувачами мережі, для здійснення оплати у криптовалюті користувачу, який не має її у своєму розпорядженні, необхідно спершу обміняти наявні у нього гроші на криптовалюту, що потребує використання онлайн-бірж або обмінників. Така конвертація валют містить додаткові комісійні витрати. Наприклад, найпопулярніша в Україні онлайн-біржа KUNA встановлює 1,5% комісійних від суми платежу при купівлі криптовалюти за фіатні гроші та 0,25% на обмін криптовалют. При здійсненні платежу у криптовалюті з рахунку на біржі встановлюються фіксовані комісійні. Таким чином, при здійсненні платежів із використанням криптовалют транзакційні витрати всередині децентралізованої платіжної системи можуть бути досить низькими та, враховуючи можливості транскордонного переміщення коштів, більш привабливими у

порівнянні із банківськими платежами. Але якщо користувачу необхідно здійснювати обмін або купувати криптовалюту для здійснення такого платежу, то додаткові комісійні витрати можуть перевищувати аналогічні витрати у банківській системі.

Ще одним важливим моментом при здійсненні платежів є спосіб підтвердження транзакцій. При використанні централізованих банківських платіжних систем банк виступає посередником і гарантом переміщення коштів між рахунками клієнтів. У ДПС визначення, чи є транзакція вірною, відбувається на основі консенсусу учасників такої системи (тобто її підтвердження здійснюється «більшістю голосів»). Понад 90% існуючих систем використовують алгоритм консенсусу PoW. Суть цього алгоритму зводиться до двох основних пунктів:

– необхідності виконання певного, досить складного і тривалого завдання;

– можливості швидко і легко перевірити результат.

Необхідність постійного розрахунку рішення робить вирішення задачі дуже ресурсномістким, у зв’язку з чим десятки тисяч комп’ютерів витрачають власні обчислювальні ресурси на виконання протоколу консенсусу і при цьому лише один з них наприкінці отримує можливість створити блок. У результаті це призводить до великих енергозатрат, що є проблемою.

Процес знаходження консесусу у блокчейні супроводжується майнінгом.

Майнінг (Mining) – це діяльність, спрямована на підтримку розподіленої платформи та створення нових блоків з можливістю отримати винагороду у формі емітованої валюти і комісійних зборів у різних криптовалютах (зокрема, у Біткоінах). По суті, майнінг – це генерація нових електронних монет, якою супроводжується процес додання нових блоків до блокчейну.

Зазвичай, майнінг зводиться до серії обчислень з перебором параметрів для знаходження хеш-суми із заданими властивостями. Ці обчислення потрібні для забезпечення захисту від повторного використання одних і тих же одиниць валюти, а зв’язок майнінгу з емісією стимулює людей витрачати свої обчислювальні потужності та підтримувати роботу мереж. ДПС напряму залежить від кількості учасників, які беруть участь у процесі майнінгу. Чим більше учасників мережі претендують отримати право на створення нового блоку, тим ціннішою є криптовалю-

та. Однак, при існуючому підході разом з цим зростає складність виконання завдання, а, отже, і шанс на отримання винагороди. Наприклад, обсяг роботи, потрібний для отримання однієї монети Біткоіна сьогодні більш ніж у 500 000 разів перевищує необхідний обсяг роботи для створення перших згенерованих монет. Тобто, чим більше людей приєднується до спільноти, тим важче стає генерувати монети. Якщо сьогодні підключитись до мережі зі звичайного комп'ютера і взяти участь у процесі майнінгу, то, найімовірніше, не вдасться заробити абсолютно нічого. Враховуючи витрати на електроенергію, подібне заняття навіть стає збитковим. Очевидно, що це стримує ріст нових майнерів, які з'являються з усе меншою інтенсивністю.

Також однією з головних проблем для ДПС залишається масштабованість. Наприклад, у рамках протоколу Bitcoin блок транзакцій обмежений розміром в 1 мегабайт і швидкість їх обробки становить приблизно сім операцій за секунду (у той же час Visa обробляє в середньому 2000 операцій за секунду). Розмір блоку впливає на кількість транзакцій, які можна додати у блок. Протокол Bitcoin передбачає, що блок формується в середньому 10 хвилин, і при збільшення активності у мережі збільшуються як комісійні, що пропонуються відправниками, так і час підтвердження транзакції вузлами мережі.

1.3 Методологічні підходи до вирішення задачі за темою дослідження

Одним з основних недоліків існуючих програмних рішень є низька швидкість проходження транзакцій.

Для оцінки продуктивності базової моделі блокчейну необхідно прийняти наступних п'ять умов.

1 Існує тільки один канал обслуговування. Незважаючи на те, що блокчейн є розподіленою системою, у підсумкову БД буде записаний лише один блок, від одного вузла.

2 Не враховується виникнення розгалужень (Forks), оскільки відмінності будуть у структурі дерева блоків, а не у послідовності транзакцій.

3 Час генерації нового блоку керується експоненційним законом (коефіцієнт коваріації для цього закону є константою, що дорівнює одиниці).

4 У блокчейн-платформі немає максимально можливого розміру блоку та обмеження за кількістю і розміром транзакції, однак існує обмеження на максимальну кількість «газу» (комісії за транзакцію), що використовується у блоці.

5 Поява нових транзакцій (заявок) підпорядковується найпростішому закону розподілу – пуассонівському.

Розглянувши ці залежності, на прикладі мережі Ethereum можна застосовувати формулу, запропоновану у роботі [11], для обрахунку середнього часу очікування заявки *w*, що залежить від інтенсивності вхідного потоку даних (формула 1).

$$w = \frac{\lambda \cdot b^2 \cdot (1 + v^2)}{2 \cdot (1 - \lambda \cdot b)},\ b = \frac{c}{d}, \tag{1}$$

де λ – інтенсивність потоку заявок;

v – коефіцієнт варіації закону розподілення середнього часу обробки однієї заявки на транзакцію;

b – середній час обробки заявки на транзакцію;

c – середній час очікування блоку;

d – кількість транзакцій у блоці.

Маючи математичні підтвердження розрахунків продуктивності системи, можна запропонувати алгоритми підвищення її пропускної здатності.

Іншим суттєвим недоліком існуючих систем є використання алгоритму консенсусу PoW. Вирішити задачу великих енергозатрат можна шляхом розробки моделі на основі протоколів, що не використовують обчислювальну здатність учасників як параметр підтримання консенсусу.

Ще одним недоліком, який часто зустрічається у сучасних ДПС є необхідність залучення третьої сторони – криптовалютних бірж для купівлі/продажу криптовалюти. Внаслідок цього вартість криптовалюти зростає через додаткові комісії для

бірж. Вирішенням цієї проблеми може бути розробка власної підсистеми, яка б займалась обміном власної криптовалюти на інші популярні цифрові валюти.

1.4 Висновки. Постановка задачі

Поява ДПС продемонструвала, що криптовалюти можуть бути ефективним інструментом інвестування. Випуск віртуальної цифрової валюти, яку потім можна використовувати як платіжний засіб усередині сервісу або компанії, виявився найпростішим та найпривабливішим способом як для залучення інвестицій, так і для інвестування. Таким чином, з'явився попит на створення платіжних систем з власними криптовалютами, «заточені» під сервісні потреби бізнесу. Основні проблеми, пов'язані з новими ДПС є наступними:

– низька швидкість проходження транзакцій (Bitcoin, Ethereum);

– необхідність залучення третьої сторони (криптовалютних бірж) для купівлі/продажу власної криптовалюти (Bitcoin, Ethereum);

– недостатня децентралізованість через використання приватних блокчейнів (Walmart, Spotify, Propy);

– зменшення зацікавленості потенційних майнерів через значне зростання складності створення блоку при збільшенні кількості учасників підтримання консенсусу (Bitcoin, Ethereum);

– функціонування системи на алгоритмі консенсусу PoW, що вимагає значних енергозатрат (Ethereum 1.0, Bitcoin).

Таким чином, спостерігається потреба у розробці оригінальної моделі ДПС, яка б вирішувала перераховані проблеми, а також в успішній імплементації розроблених моделей та механізмів для забезпечення безпечного, прозорого та ефективного процесу створення власної криптовалюти.

При детальному ознайомленні із предметною областю було сформувані наступні вимоги до віртуальної валюти:

– забезпечення емісії (емісія – це випуск в обіг нових грошових знаків і платіжних засобів, що викликає збільшення грошової маси);

– взаємозамінність валюти (взаємозамінність валюти полягає у тому, що всі її одиниці є еквівалентними);

– забезпечення обігу – у самій валюті мають бути реалізовані можливості передачі певної кількості валюти безпосередньо іншому власнику або надання права розпоряджатись цією валютою (allowance);

– незалежність валюти (функціонування валюти має бути незалежним від будь-яких інших компонентів системи);

– децентралізованість (після випуску валюти ніхто, включно зі створювачем валюти, не може ніяк впливати на її характеристики, закладені при створенні).

Також було визначено наступні функції, які повинна забезпечувати ДПС:

– створення віртуальної валюти;

– купівля валюти за інший віртуальний актив;

– реєстрація та авторизація користувачів;

– взаємодія з системою напряму, без посередників у вигляді веб-інтерфейсу;

– надання інформації про історію транзакцій користувача, баланс, інформації про валюту (розмір емісії, поточна ціна, назва).

Відповідно для цього слід провести глибокий системний аналіз процедури залучення криптовалютних інвестицій, запропонувати оригінальні ефективні методи та способи вирішення наявних проблем та розробити програмний комплекс для ДПС з обігу власної криптовалюти.

Таким чином, актуальність теми роботи полягає у необхідності розробки повнофункціональної ДПС, яка б дозволяла здійснювати фінансові перекази цифрової валюти, відповідала стандартам безпеки та була незалежною від банківських регуляторів.

Об’єктом дослідження є процеси функціонування ДПС та програмно-технічна база, необхідна для забезпечення обігу цифрової валюти у рамках ДПС.

Предмет дослідження – моделі, методи та механізми створення безпечної, прозорої та ефективної ДПС, яка б забезпечувала процес обігу цифрової валюти.

Мета дослідження – розробка програмного комплексу для децентралізованої платіжної системи з обігу власної криптовалюти.

Відповідно до поставленої мети задачі дослідження є наступними:

– проаналізувати специфіку функціонування ДПС;

– дослідити процедури залучення криптовалютних інвестицій та обґрунтувати необхідність розробки системи для обігу криптовалют;

– провести аналіз існуючих моделей та методів у галузі криптовалютних інвестицій, виділити невирішені проблеми;

– визначити основні вимоги до ДПС та функції, які вона має виконувати;

– удосконалити моделі та методи організації процесу функціонування ДПС, які б вирішували наявні проблеми у галузі криптовалютних інвестицій;

– виконати проектування ПС на основі розроблених методів та алгоритмів;

– виконати програмну реалізацію прийнятих рішень;

– провести тестування та практичну апробацію отриманих результатів;

– проаналізувати отримані результати та сформувати рекомендації щодо доцільності впровадження результатів дослідження.

Таким чином, у розділі проаналізована предметна область, наявні методи та засоби у галузі криптовалютних інвестицій, виділені невирішені частини загальної проблеми та описані методологічні підходи до їх вирішення, а також виконана розгорнута постановка задачі для подальшого дослідження.

2 КОНЦЕПЦІЇ, МОДЕЛІ ТА МЕТОДИ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ

2.1 Концепції розробки децентралізованої платіжної системи

Традиційні ДПС складаються з двох архітектурних компонентів. Основна логіка роботи системи працює на блокчейн-платформі (наприклад, на розумних контрактах), а для забезпечення зручних інтерфейсів розробляться веб-частина, до складу якої найчастіше входять наступні інструменти: оглядач блоків та транзакцій для показу статистичної та службової інформації про платіжну систему; відділ адміністрування, де здійснюється керування платформою; криптогаманець, за допомогою якого кінцевий користувач здійснює операції надсилання криптовалютних коштів чи отримує інформацію про їх отримання [12]. При такій формі організації системи існує очевидна проблема − після закінчення емітування монет у мережі користувачі можуть отримати криптовалюту, лише купивши її в інших учасників системи або на криптовалютних біржах. Це призводить до появи небажаних комісій та додаткових ризиків для користувачів, які вимушені користуватись сторонніми додатками для придбання криптовалюти.

Вирішенням описаної проблеми може стати розробка власної обмінної платформи в комплексі однієї платіжної системи. Таким чином, користувач матиме вибір − здійснювати переказ через зовнішні біржі на ринку чи скористатись офіційною обмінною платформою. В той час, як на офіційній обмінній платформі ціна на криптовалюту буде вищою за ринкову, користувачі, які нею користуватимуться, будуть впевнені у безпеці проведення своїх операцій, оскільки їм не потрібно покладатись на сторонні платформи. На рисунку 2.1 наочно продемонстровано загальну схему платформи та те, як компонент обмінної системи інтегрується в неї.

Розглянемо детальніше компоненти на представленій схемі. Очевидно, що основним її компонентом є блокчейн-система, яку використовують інші частини платформи для виконання операцій. Блокчейн-система реалізовує головний функціонал системи − збереження даних про криптовалюту та користувацькі рахунки, емітування монет, опрацювання переказів між користувачами. Інші компоненти системи лише звертаються до методів розумних контрактів для отримання даних чи виконання певних операцій.

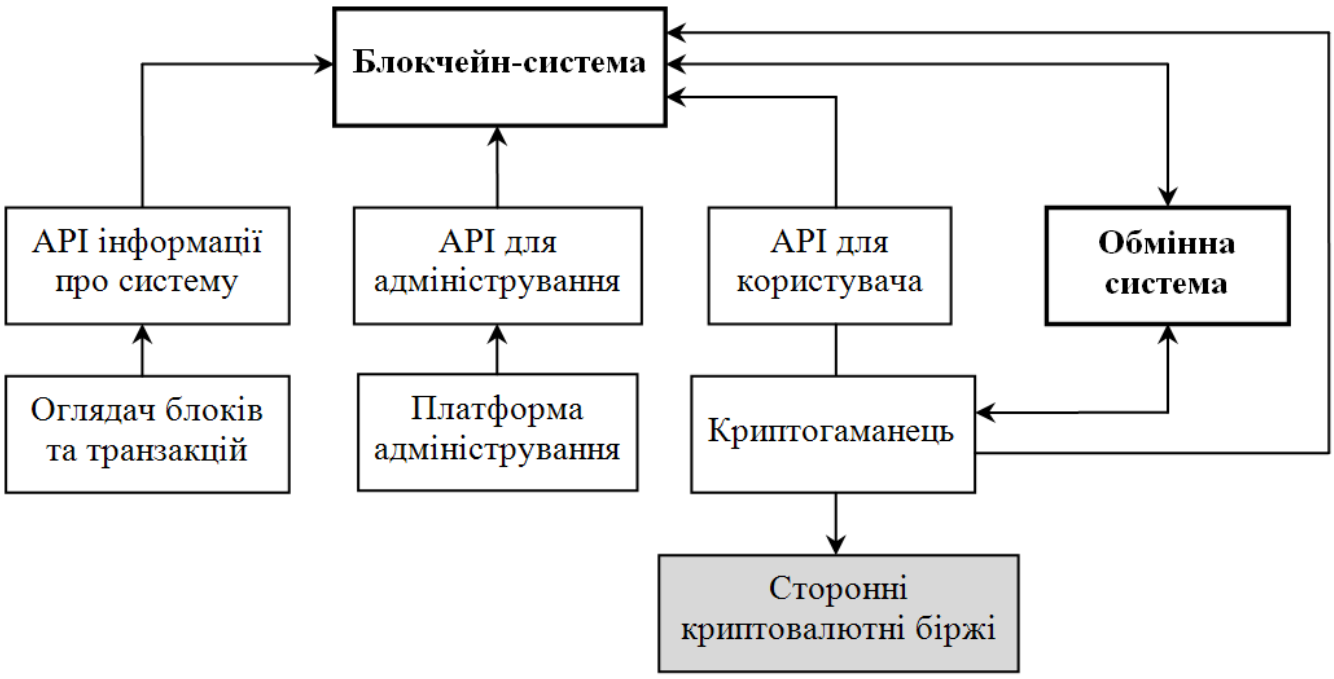


Рисунок 2.1 − Взаємодія компонентів системи

В той час, як операції для отримання даних з блокчейну відбуваються за допомогою звичайних методів читання бази даних смарт-контрактів, які миттєво повертають результат, виконання операцій на модифікацію існуючих даних (таких, наприклад, як перекази між рахунками) відбуваються за визначеним алгоритмом (рисунок 2.2). Усі транзакції в системі зберігаються у спеціальних структурах даних − блоках. Під час роботи системи через деякий визначений часовий інтервал у блокчейн додається новий блок. Цей блок може бути порожнім або містити інформацію про деяку кількість транзакцій, обмежену розміром блоку. Кожний наступний блок зберігає посилання на попередній, таким чином формуючи єдиний нерозривний ланцюг. При додаванні нового блоку до ланцюга транзакції, що містяться в ньому, почергово виконуються на розумних контрактах [13].

Очевидно, що з таким алгоритмом опрацювання транзакцій неможливо здійснювати горизонтальне масштабування системи. Незважаючи на те, що блокчейн є розподіленою системою, у підсумкову БД записується лише один блок, від одного вузла, тому що опрацювання блоків може здійснювати тільки один канал обслуговування. Оскільки збільшення кількості каналів обробки неможливе, для підвищення пропускної здатності системи в роботі запропоновано наступні методи оптимізації.

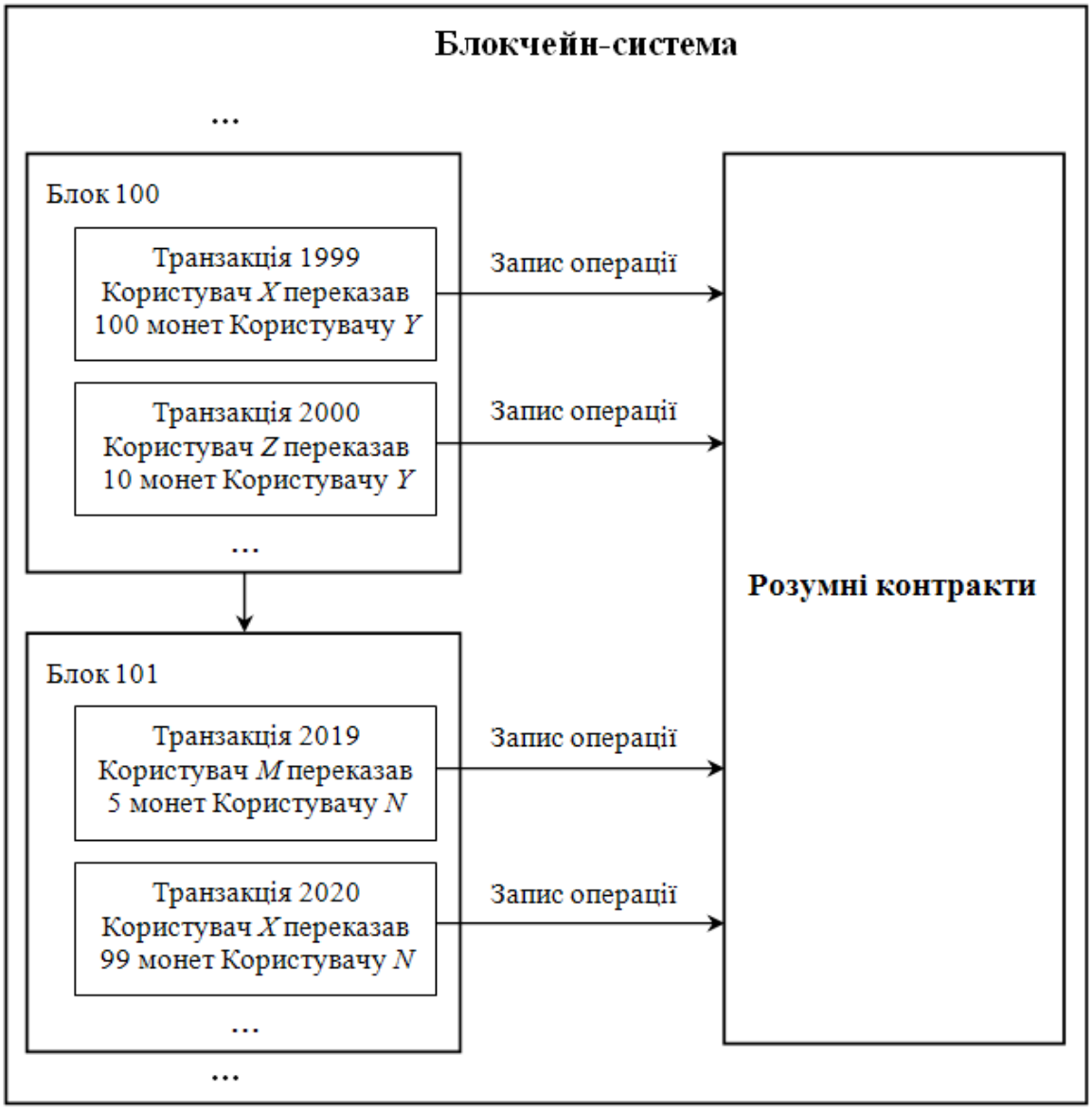


Рисунок 2.2 − Опрацювання транзакцій всередині блокчейн-системи

Використання динамічного розміру блоку

Кожний блок у системі має фактичний та максимальний розмір. Фактичний розмір блоку відповідає об’єму транзакцій, які в ньому знаходяться. Максимальний розмір блоку визначається системою під час її старту. Передбачивши можливість збільшення максимального розміру блоку вже після запуску мережі, можна масштабувати її пропускну здатність: при збільшенні навантаження на систему значення максимального розміру блоку буде зростати, а при зменшенні навантаження – зменшуватись. Таким чином, з’явиться змога опрацьовувати більшу кількість транзакцій за той самий часовий період. Реалізація цього функціоналу полягатиме у підтриман-

ні консенсусної зміни конфігурацій. Створювачу блоків має бути надана можливість змінювати максимальний розмір для свого блоку, але не більше ніж на 1% від розміру попереднього блоку. Таким чином, значення максимального розміру блоку у системі відповідатиме навантаженню на систему.

Перекази «один до багатьох» («One-to-Many»)

Також, для того, щоб зменшити навантаження на мережу, доцільно розробити функціонал виклику трансферу криптовалюти, при якому в якості вхідних аргументів методу смарт-контракту можна було б передати список адрес отримувачів та необхідні суми переказу кожному з отримувачів. Таким чином, при здійсненні переказу на значну кількість адрес, користувач зможе це зробити всередині лише однієї транзакції, що зменшить навантаження на мережу.

У процесі застосування технології блокчейн виникає також питання підбору ефективного протоколу консенсусу. Консенсус блокчейна полягає в тому, що всі вузли підтримують однаковий розподілений реєстр. У традиційній архітектурі програмного забезпечення консенсус не є проблемою через існування центрального сервера, з яким узгоджуються інші вузли. Однак, у розподіленій мережі (такій, як блокчейн) кожний вузол є і хостом, і сервером, і, щоб досягти консенсусу, йому потрібно обмінюватися інформацією з іншими вузлами. Інколи деякі вузли можуть працювати у режимі офлайн. Крім того, можуть з’явитись деякі шкідливі вузли, які будуть негативно впливати на процес консенсусу і, навіть, можуть зашкодити йому. Тому потрібний протокол, який не допустить виникнення подібних ситуацій і мінімізує шкоду від шкідливих вузлів таким чином, щоб вони не впливали на кінцевий результат консенсусу.

Аналіз предметної області показав, що близько в 90% існуючих децентралізованих системах використовується алгоритм консенсусу PoW [14]. Однак, значним недоліком PoW-алгоритму є те, що його функціонування потребує постійних значних затрат електроенергії на процес створення блоків. Цей недолік яскраво видно на прикладі Біткоіна, який реалізує вказаний алгоритм. За один рік на функціонування мережі йде більше електроенергії ніж використовує Швейцарія за цей ж період. Проблему значних енергозатрат можна вирішити шляхом розробки

моделі на основі протоколів, що не використовують обчислювальну здатність учасників як параметр підтримання консенсусу. У роботі [6] автори здійснили порівняльний аналіз найпопулярніших алгоритмів консенсусу, розглянувши методи обчислення значень основних характеристик алгоритмів: відмовостійкість, ресурсоємність, масштабованість та придатність для публічних мереж. На основі результатів цього аналізу сформовано порівняльну таблицю 2.1.

Таблиця 2.1 – Характеристика різних алгоритмів консенсусу

Характеристика	PoW	PoS	PBFT	Ripple
Тип алгоритму	Ймовірнісно-кінцевий	Ймовірнісно-кінцевий	Абсолютної остаточності	Абсолютної остаточності
Відмовостійкість	50%	50%	33%	20%
Ресурсоємність	Висока	Середня	Низька	Низька
Масштабованість	Добра	Добра	Погана	Погана
Придатність для публічних мереж	Придатний	Придатний	Не придатний	Придатний

Як можна бачити з таблиці 2.1, серед алгоритмів, придатних для публічної мережі, алгоритми PoW і PoS відрізняються лише за показником ресурсоємності. При цьому PoS є менш ресурсоємним, тому він є кращим вибором для публічного блокчейну. Вибір цього алгоритму консенсу дозволить зменшити ресурсоємність системи та забезпечить необхідними інструментами персоналізації.

Як було зазначено у підрозділі 1.2, робота ДПС напряму залежить від кількості учасників, які беруть участь у процесі майнінгу. Чим більше учасників мережі претендують отримати право на створення нового блоку, тим ціннішою є криптовалюта. Однак, при існуючому підході багато учасників мають дуже низькі шанси отримати ці права. І хоча загальна тенденція – чим більше людей приєднується до спільноти, тим важче стає генерувати монети – є природньою, проблемою залишається те, що це стримує збільшення кількості нових майнерів (вони з’являються з усе меншою інтенсивністю). Новим майнерам з непотужним обладнанням дуже складно отримати право на отримання блоку, а, отже, вони ризикують працювати

собі у збиток [15]. Цю проблему можна вирішити на рівні системи – достатньо визначити новий, справедливий спосіб розподілення винагороди за створення блоку, при якому виногорода розподілялася б рівномірно між учасниками системи відповідно до ресурсів, які вони вклали у роботу апаратного забезпечення для підтримання консенсусу. Рівномірний розподіл емісії між учасниками дозволить уникнути ситуацій, у яких деякі учасники не отримували взагалі нічого, і, таким чином, мотивуватиме нових учасників приєднуватись до підтримання консенсусу системи.

2.2 Моделі та методи розробки децентралізованої платіжної системи

Розглянемо детальніше методи, описані у попередньому розділі та, базуючись на них, розробимо нову модель системи, а також доведемо, що розроблена система матиме кращі показники ефективності, ніж традиційна.

Спершу розглянемо механізми функціонування власної обмінної системи. Така система, за своїм призначенням, має здійснювати два типи операцій: купівля цифрової валюти системи за іншу популярну цифрову валюту та продаж валюти системи в обмін на іншу цифрову валюту. Метою інтеграції обмінної системи є позбавлення користувача необхідності використовувати для цього сторонні рішення (сторонні криптовалютні біржі) та зменшити розмір комісій, яку витрачає користувач на купівлю/обмін криптовалюти.

Детальний механізм взаємодії обмінної системи та користувача представлений на рисунку 2.3.

У традиційній моделі, де відсутня обмінна система, користувач може отримати криптовалюту, придбавши її на зовнішній біржі (чи обміннику). При цьому користувач використовує ці сервіси на свій страх і ризик, оскільки вони не мають прямого стосунку до платіжної системи і можуть виявитись шахрайськими. Також користувач змушений платити додаткові комісії для цих систем, які платіжна система не може контролювати.

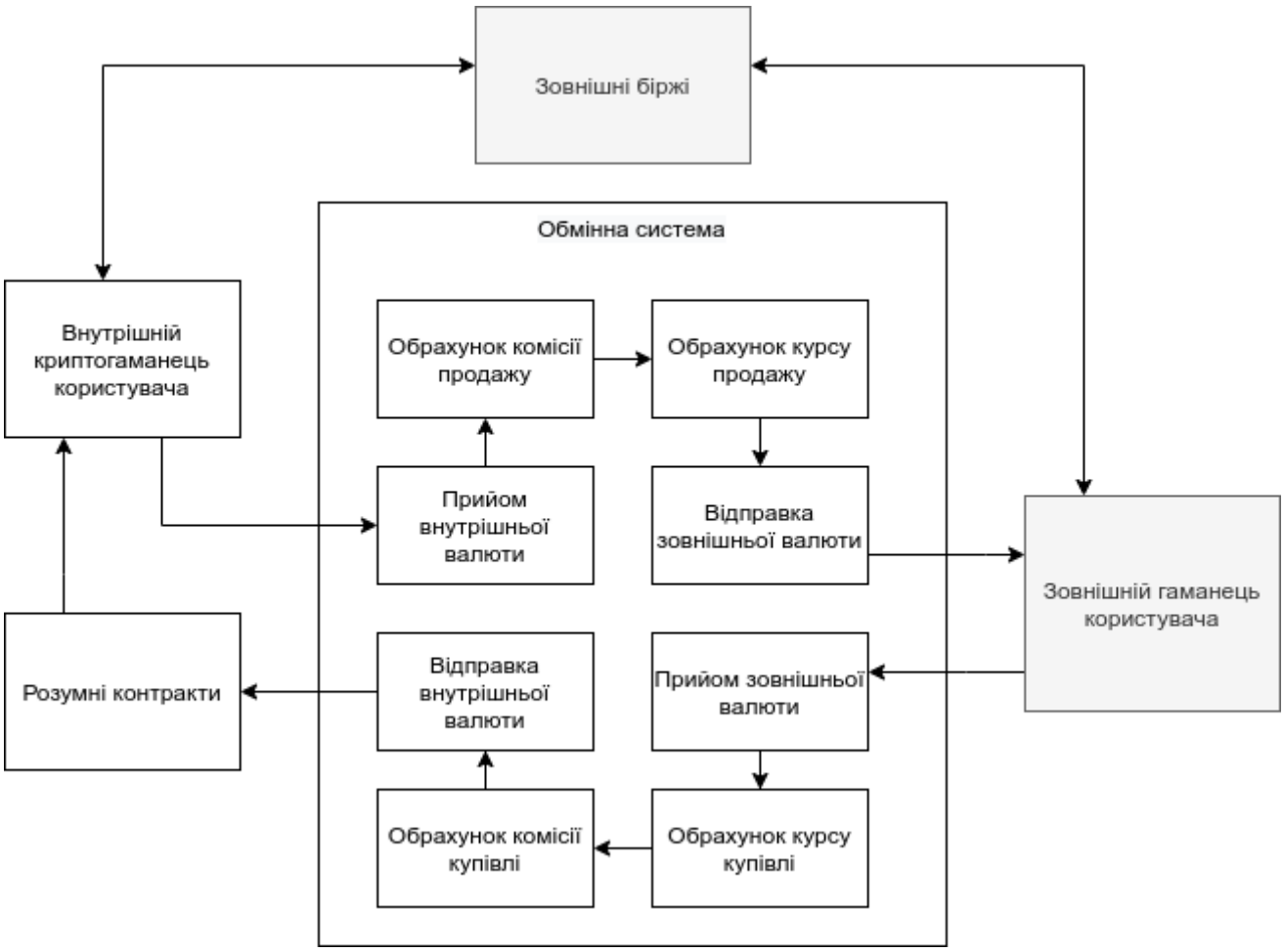


Рисунок 2.3 − Взаємодія обмінної системи та користувача

У моделі з інтегрованою обмінною системою користувач може переказати свої кошти прозорим способом всередині платформи. Це убезпечить його від додаткових ризиків та комісій. Для отримання внутрішньої криптовалюти користувачу потрібно відправити кошти зі свого зовнішнього гаманця на рахунок обмінної системи. Після цього відбудеться розрахунок внутрішнього курсу валюти та внутрішньої комісії (яка не є способом заробітку, а лише необхідна для покриття вартості проведення транзакції із зарахуванням внутрішньої валюти). В результаті сума буде конвертована у внутрішню валюту та зарахована з рахунку розумного контракту на рахунок користувача. Схожим чином ситуація відбуватиметься у зворотній бік: користувач повинен відправити кошти зі свого внутрішнього криптогаманця на рахунок обмінної системи і, після конвертації та вирахування комісійних витрат, отримати зовнішню криптовалюту на свій зовнішній рахунок.

Також запропоновано використовувати динамічний розмір блоку задля оптимізації пропускної здатності системи. Переваги використання такого методу можна обґрунтувати, скориставшись формулою 1. При забезпеченні збільшення розміру блоку, ми, як результат, отримуємо можливість розмістити у блоці більшу кількість транзакцій (d). Взявши інші параметри рівняння 1, як сталі значення, для порівняння традиційної та покращеної моделі системи, виявимо, що середній час обробки заявки на транзакцію (c) є обернено пропорційним до кількості транзакцій у блоці, а середній час очікування заявки (w), у свою чергу, є прямо пропорційним до часу її обробки. Таким чином отримаємо, що, при рівності інших параметрів, час очікування заявки у покращеній моделі буде пропорційно зменшуватись зі збільшенням максимального розміру блоку.

Реалізація трансферу криптовалюти «One-to-Many» («один-до-багатьох») дозволить користувачу відправляти всередині однієї транзакції декілька різних переказів на різні адреси. Це, у свою чергу, призведе до зменшення об’єму операцій, які необхідно здійснити над базою даних порівняно з аналогічним трансфером, але проведеним у декілька транзакцій (рисунок 2.4).

Очевидно, що цей спосіб містить обмеження: всередині транзакції може бути здійснено перекази на інші адреси лише з одного рахунку користувача. Проте, як видно з рисунка 2.4, такі перекази дозволять зменшити кількість операцій з розумними контрактами та вмістити більше транзакцій в один блок. Це, у свою чергу (як було доведено раніше), дозволить збільшити пропускну здатність системи.

Використання алгоритму консенсусу PoS для учасників системи полягає у необхідності доказу зберігання певної кількості криптовалюти. При використанні цього методу алгоритм з більшою ймовірністю вибере для підтвердження чергового блоку у ланцюжку обліковий запис з великою кількістю коштів на рахунку. Використання цього методу, як альтернативи методу PoW, у якому більшу ймовірність підтвердження блоку має обліковий запис з великими обчислювальними потужностями, дозволить звести до мінімуму всі витрати, пов’язані з підтриманням консенсусу. Аргументи, що підтверджують спроможність методу протистояти атакам на консенсус:

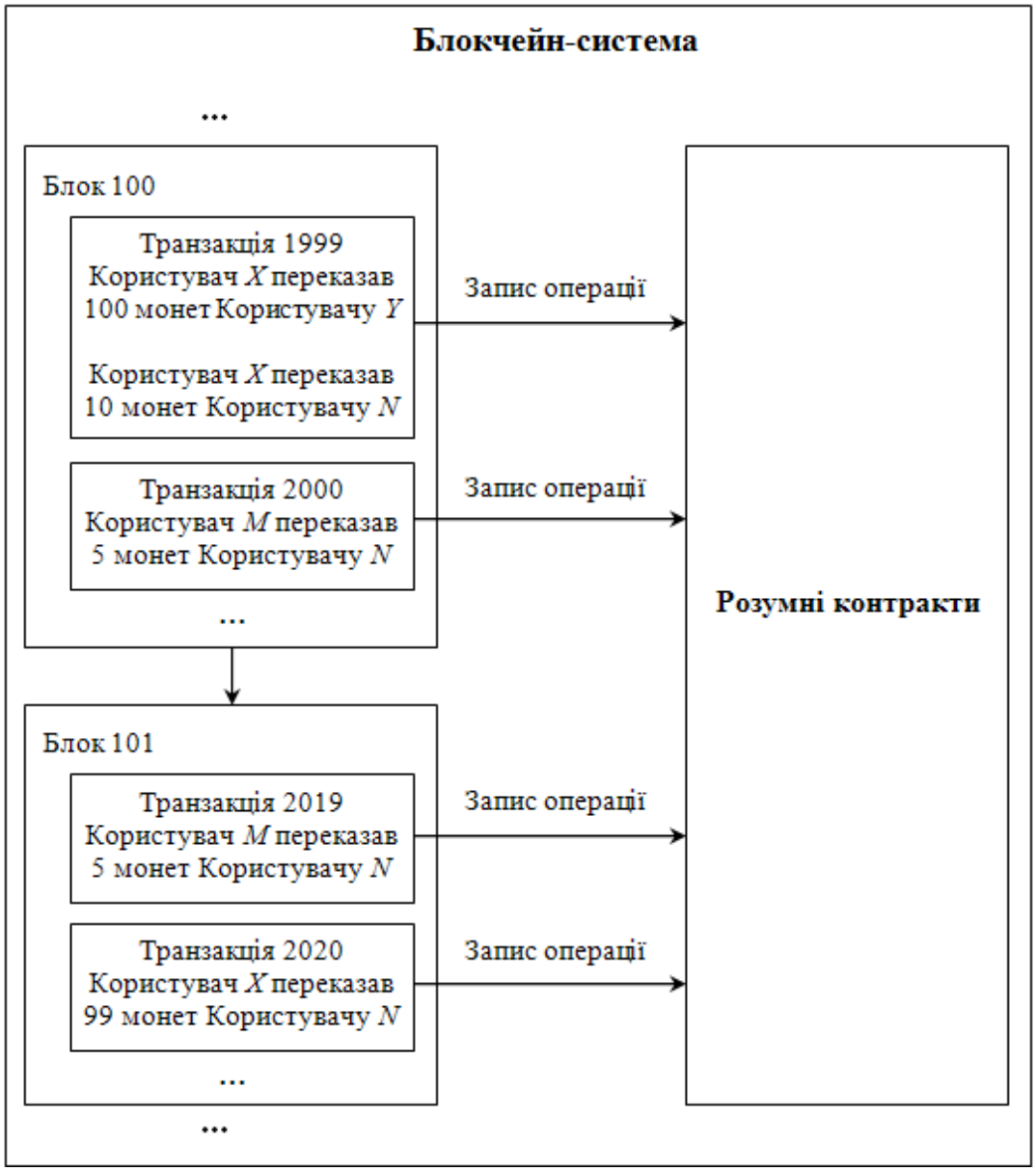


Рисунок 2.4 – Проходження переказів у системі всередині однієї транзакції

– для проведення атаки 51% потрібно багато коштів (атакуючому буде просто дорого виконати атаку);

– якщо у атакуючого знайдеться багато коштів, то він сам постраждає від атаки, оскільки це порушить стійкість криптовалюти.

Розглянемо спосіб оптимізації розподілення винагороди між учасниками пошуку консенсусу. У традиційній моделі розподілення винагороди між учасниками відбувається випадковим чином. Перед створенням блоку алгоритм обирає одного з учасників консенсусу, наприклад «А», і віддає йому право на додавання цього блоку до ланцюга; при цьому учасник «А» отримує винагороду. У довгостроковій

перспективі, за умови постійної участі учасника «А» у підтриманні консенсусу та рівномірному розподіленню результатів алгоритму серед учасників, «А» буде отримувати винагороду у тому обсязі, який пропорційний його участі. Однак, якщо на деякому блоці учасник захоче припинити брати участь у консенсусі, кількість винагороди, яку він до цього отримав, може не бути співставною з його участю. Ця ж проблема стосується ситуації, коли новий учасник приєднався до підтримання консенсусу, але має пройти певний період часу, перш ніж він зможе отримати винагороду. Цю проблему можна вирішити шляхом розподілення винагороди за кожний блок рівномірно між усіма учасниками, що брали участь у його створенні, відповідно до їхнього вкладу у підтримання консенсусу системи (рисунок 2.5).

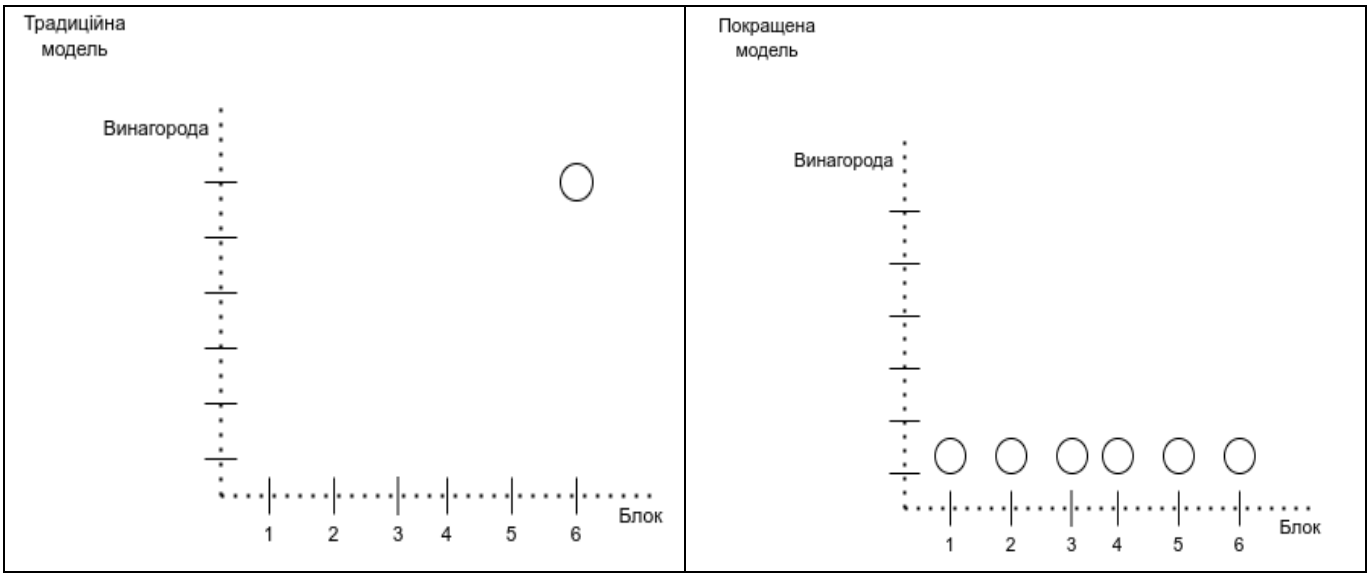


Рисунок 2.5 – Порівняння отримання винагороди у традиційній та покращеній моделях

Як видно на графіках, представлених на рисунку 2.5, і в одному, і в другому випадку учасник сукупно отримує однакову суму винагороди. Однак, у традиційній моделі цю суму він отримує одноразово, до цього моменту працюючи «просто так», а у покращеній моделі він отримує відсоток від участі у консенсусі при створенні кожного блоку. Таким чином, учаснику це надає наступні переваги:

– на якому б блоці учасник не припинив брати участь у консенсусі, сума отриманої ним винагороди завжди буде пропорційною його участі у консенсусі;

– учасник, який приєднався до підтримання консенсусу, відразу ж отримує дивіденди (винагороду).

2.3 Організація захисту даних та безпеки системи

Оскільки розроблювана ПС має відповідати високим вимогам безпеки, потрібно передбачити механізми захисту користувачів від несанкціонованого доступу задля безпеки коштів.

Так як платформа складається з декількох частин, потрібно розглядати безпеку системи у цілому, як безпеку окремих компонентів. Так, наприклад, практично неможливо забезпечити високий рівень безпеки при зберіганні авторизаційних даних до криптогаманця користувача у централізованому сховищі. Виходячи з цього, слід розглядати безпеку блокчейн-частини додатку ізольовано від інших частин; жоден із додатків веб-частини не повинен ні в якому вигляді зберігати чи обробляти ключі доступу користувача. Окрім цього, для повноцінного користування системою користувачу абсолютно непотрібно користуватись веб-частиною – вона слугує лише допоміжним інструментом. Тому у системі варто реалізувати два рівні захисту:

– перший рівень захисту – на рівні веб-частини, який дозволить користувачу отримати дані його особистого акаунту, персоналізувати надходження інформації від системи, створити заявки на обмін тощо;

– другий рівень захисту, який працює напряму із блокчейном та дозволяє проводити операції безпосередньо з коштами.

Перший рівень не потребує використання складних криптографічних алгоритмів, оскільки, навіть при втраті доступу до акаунта або при отриманні несанкціонованого доступу третіми особами, це не потягне за собою жодних економічних наслідків. Тому для першого рівня захисту достатньо реалізувати стандарті механізми авторизації/аутентифікації через логін та пароль, хеш якого зберігатиметься у БД.

Вимоги до другого рівня захисту є жорсткішими, тому його варто реалізо-

вувати лише на рівні звернень до блокчейну. Для цього зручно скористатись ECDSA-алгоритмом, який реалізований Ethereum. Так, для функціонування цього алгоритму Ethereum генерує два ключі – публічний та приватний. Публічний ключ може слугувати одночасно ідентифікатором відправника та адресою його гаманця, а приватний – виконувати роль пароля. При створенні заявки на відправлення транзакції, яка може бути сформована офлайн, користувачу необхідно підписати її своїм приватним ключем. У результаті заявка разом із цифровим підписом відправляється у систему, де відбувається верифікація підпису. Звіривши адресу відправника, звідки потрібно списати кошти з публічним ключем, отриманим від процесу перевірки справжності підпису, система вирішує, чи має право користувач здійснювати цю транзакцію. Таким чином, у системі ніде не потрібно зберігати дані про користувача. Згенерувавши ключі, користувачу не потрібно надсилати приватні дані для системи. Тому це виключає випадки, коли повідомлення можуть бути перехоплені, і стороння особа отримає змогу дізнатись, наприклад, деякі секретні дані чи коли хтось може «зламати» систему і, таким чином, отримати несанкціонований доступ до акаунту. Всі секретні дані зберігаються тільки на стороні клієнті і безпека його гаманця стосується лише питань безпеки його пристрою для зберігання даних.

2.4 Висновки

У розділі здійснено аналіз моделей та методів вирішення проблем у галузі криптовалютних інвестицій. Наприклад, при здійсненні платежу у криптовалюті користувачу необхідно спочатку здійснити обмін або купівлю криптовалюти на криптовалютній біржі, де додаткові комісійні витрати можуть перевищувати аналогічні у банківській системі. Цю проблему запропоновано вирішити шляхом розробки власної обмінної платформи. Було представлено наочну схему структури традиційної платіжної системи та те, як обмінна платформа інтегрується в неї. Користувачі такої системи будуть впевнені ув безпеці проведення своїх операцій, оскільки їм не потрібно покладатись на сторонні криптовалютні біржі.

Окрім цього, однією з головних проблем для ДПС залишається масштабованість. Оскільки опрацювання транзакцій у блокчейні може здійснювати лише

один канал обслуговування, це призводить до необхідності оптимізації роботи системи іншими способами. Для збільшення пропускної здатності системи у розділі запропоновано наступні методи оптимізації системи: використання динамічного розміру блоку та імплементація переказу типу «один до багатьох». Динамічний розмір блоку дозволить на рівні алгоритму збільшувати чи зменшувати максимальний розмір блоку, в залежності від кількості транзакцій, що очікують виконання, забезпечивши таким чином необхідний рівень пропускної здатності. Імплементація переказів «один до багатьох» передбачає розробку нових методів розумних контрактів для здійснення переказу від одного користувача на значну кількість адрес всередині лише однієї транзакції, що дозволяє зменшити навантаження на мережу.

Також встановлено, що більше ніж 90% існуючих платіжних систем працюють на неефективних алгоритмах консенсусу, які вимагають значних енергозатрат. Цю проблему вирішено шляхом розробки моделі на основі протоколів, які не використовують обчислювальну здатність учасників як параметр підтримання консенсусу, а саме алгоритму PoS, суть якого полягає у необхідності доказу зберігання певної кількості криптовалюти.

З’ясовано, що у традиційній моделі, якщо на деякому блоці учасник системи захоче припинити брати участь в підтриманні консенсусу, то кількість винагороди, яку він до цього отримав, може не бути співставною з його участю у консенсусі. Ця ж проблема стосується ситуації, коли новий учасник приєднався до підтримання консенсусу, але має пройти певний період часу, перш ніж він зможе отримати винагороду. Проблему запропоновано вирішити шляхом розподілення винагороди за кожний блок рівномірно між усіма учасниками, що брали участь у його створенні, відповідно до їхнього вкладу у підтримання консенсусу системи.

Також зазначено, що у системі варто реалізувати два рівні її захисту: на рівні веб-частини та на рівні, який працює напряму із блокчейном.

Наступним етапом роботи є опис можливостей використання розроблених моделей і методів за допомогою алгоритмів, опис архітектурного та компонентного дизайну ПС та її детальних проектних рішень.

3 АЛГОРИТМИ ТА ТЕХНОЛОГІЇ ВИРІШЕННЯ ЗАДАЧІ

3.1 Алгоритми вирішення задачі

Визначивши методи вирішення наявних у галузі проблем, перейдемо до опису алгоритмів, які їх використовуватимуть.

Розглянемо детальніше процедуру обміну зовнішньої криптовалюти на цифрову валюту розроблюваної платіжної системи. Виконання такого обміну складатиметься з наступних етапів:

– формування заявки на обмін;
– підтвердження вхідної транзакції;
– розрахування комісії та курсу;
– надсилання криптовалюти на рахунок користувача.

Формування заявки на обмін

Користувач, який бажає обміняти зовнішню криптовалюту на криптовалюту системи, спочатку ініціює створення заявки, де він вказує свою криптографічну адресу, на яку він бажає отримати криптовалюту. При цьому в обмінній системі відбувається створення гаманця для отримання коштів спеціально під заявку. Система генерує ключі доступу до гаманця, повідомляє користувачу публічну адресу гаманця та встановлює автоматичного слухача з визначеним таймером до новоствореного гаманця. Допоки не спливе час таймера (зазвичай, не більше декількох годин), слухач буде спостерігати за змінами на рахунку гаманця. Якщо таймер закінчився, але транзакції так і не відбулось, слухач відключається, а заявка переходить у статус «невиконана». Якщо протягом функціонування таймера слухач виявляє вхідну транзакцію на рахунок, він змінює статус заявки на «чекає підтвердження», вимикає таймер та відключається сам.

Підтвердження вхідної транзакції

У випадку, коли вхідна транзакція була розпізнана слухачем, вона повинна бути додатково підтверджена. Справа в тому, що у блокчейні можливою є ситуація так званої «побічної гілки». Це ситуація, коли вузол, до якого звертається користувач, повертає інформацію про деякий блок, як про включений у ланцюг, але

водночас існують інші вузли, які включили у свій ланцюг зовсім інший блок. Таким чином, виникає розсинхронізація блокчейну на вузлах, і на деякому моменті їх необхідно синхронізувати. Тоді валідним визнається ланцюг з більшою довжиною, і ті вузли, які до цього визнавали неправильний ланцюг, мусять переписати історію останніх блоків відповідно до нових даних. Тому у блокчейні не можна довіряти транзакціям, які щойно виконались; потрібно дочекатись достатньої кількості блоків у ланцюгу (зазвичай, 3-4) для того, щоб упевнитись, що поточна гілка блоків не виявиться побічною і не буде «відкочена».

Розрахування комісії та курсу

Після отримання та підтвердження вхідної транзакції обмінна система дістає з БД актуальний відсоток комісії за переказ та, користуючись зовнішніми API, актуалізує поточний курс криптовалют. На основі отриманих даних формується значення суми криптовалюти системи, яка буде зарахована на адресу користувача, вказану ним на першому етапі.

Надсилання криптовалюти на рахунок користувача

Після всіх підтверджень та обрахувань система звертається до розумного контракту, вказуючи суму переказу. Ця кількість криптовалюти зараховується на рахунок користувача. Інформація про транзакцію записується у заявку, а сама заявка змінює статус на «опрацьована».

В будь-який момент користувач може скасувати заявку; при цьому слухачі, прикріплені до гаманця, будуть вимкнені, і обмінна система припинить реагувати на нові вхідні транзакції для поточної заявки.

У разі отримання коштів користувачу надаються гарантії в отриманні відповідної кількості криптовалюти системи.

Розглянемо детальніше процедури оптимізації пропускної здатності системи:

– динамічне збільшення розміру блоку;

– імплементацію переказу «один до багатьох».

Динамічне визначення розміру блоку після старту системи може бути досягнуто шляхом конфігурування процедури підтримання консенсусу. До стандартного алгоритму, коли створювач генерує блок визначеного розміру,

необхідно додати можливість збільшити розмір блоку, проте лише в ситуації, коли створювач може заповнити його повністю транзакціями. В іншому випадку, коли у збільшенні блоку немає необхідності через недостатню кількість транзакцій, які знаходяться у стані очікування, розмір блоку повинен, навпаки, зменшуватись. Таким чином, досягається стан системи, у якому її пропускна здатність адаптується до зовнішнього навантаження.

Алгоритм переказів «один до багатьох» полягатиме в оптимізації розміру транзакцій. Розмір транзакції визначається загальною кількістю операцій, яка буде здійснена під час неї (додання даних, читання даних, перезаписування даних тощо). Транзакції користувача потрібно розглядати як окремий набір операцій над блокчейном. Відтак, якщо об'єднати декілька транзакцій користувача в одну, то можна позбутись частини операцій, і, отже, зменшити сукупний розмір транзакції. Наприклад, замість послідовних операцій віднімання рахунку при кожній транзакції, операцію віднімання можна здійснити лише раз, а зарахування для кожного отримувача здійнювати вже окремо. Таким чином, вдасться оптимізувати розмір транзакцій, а, отже, зменшити навантаження на систему.

Наступним етапом є вибір механізму підтримання консенсусу у мережі. У традиційній моделі, де використовується алгоритм консенсусу PoW, учасникам консенсусу потрібно постійно проводити складні обчисленння, що вимагають значних енергозатрат. Як було показано у першому розділі, систему можна оптимізувати, застосовуючи алгоритм PoS, який не використовує обчислювальну здатність учасників як параметр підтримання консенсусу. Алгоритм PoS заснований на необхідності доказу зберігання певної кількості коштів на рахунку. При використанні цього методу алгоритм з більшою ймовірністю вибере для підтвердження чергового блоку у ланцюжку обліковий запис з великою кількістю коштів на рахунку.

Далі розглянемо алгоритм рівномірного розподілу винагороди за створення блоку. У традиційній моделі нагорода за блок дістається одному учаснику, однак консенсус розподіляє право на створення блоку рівномірно між усіма, таким чином урівнюючи їхні шанси відповідно до участі в алгоритмі. Однак, такий підхід володіє очевидним мінусом, який полягає в нерівномірному розподілі винагороди в часі.

Рішенням може бути можливість об’єднання довільної кількості створювачів блоків у групу. Таким чином, абстрактно можна розглядати цю групу як єдиного створювача блоків, який буде отримувати шанси відповідно до сукупного вкладу всіх учасників групи. Винагорода всередині групи буде рівномірно розподілятись між її учасниками залежно від їхнього вкладу у підтримання консенсусу. У підсумку, всі учасники групи будуть отримувати відсоток від винагороди після створення кожного блоку. А для користувачів, які не бажають брати участь в групі, залишиться можливість брати участь в алгоритмі консенсусу як окремі учасники.

Після опису алгоритмів, на яких базуватимуться частини системи, щодо яких розроблялись методи вирішення наявних у галузі криптовалютних інестицій проблем, можна перейти до більш повного та широкого опису загальних вимог до ДПС.

3.2 Визначення вимог до програмної системи

На основі аналізу предметної області, розроблених методів та алгоритмів, можна визначити та описати вимоги до створюваної ПС. Спершу опишемо специфікації на рівні бізнес-вимог.

Бізнес-вимоги описують мету, яку необхідно досягти в результаті розробки. Основною метою самої роботи є успішне та ефективне функціонування платіжної системи, яка дозволить користувачам здійснювати внутрішні перекази коштів, а також обмінювати зовнішні фінансові ресурси на новостворену криптовалюту, яка існуватиме як самостійний актив.

Оскільки встановлена мета – розробити повнофункціональну ДПС, то, окрім оптимізації, її слід забезпечити всіма основними функціями, яких потребує ринок.

Сформуємо вимоги користувачів системи. В системі існують три ролі кінцевих користувачів: інвестори, адміністратори та майнери. Оскільки з ДПС будуть взаємодіяти користувачі різного рівня комп’ютерної грамотності, слід забезпечити максимально інтуїтивний процес роботи зі зручним, зрозумілим інтерфейсом. Для цього користувачів необхідно забезпечити повним пакетом необхідних інструмен-

тів, які б повністю задовольнили їхні інформаційні потреби. Користувачі повинні мати доступ до свого криптогаманця, обмінної системи та загальної інформації про платформу. Адміністратори повинні мати доступ до детальної статистики, історії транзакцій користувачів та кабінету адміністрування платформи. Майнерам необхідно забезпечити зручні та ефективні механізми участі у підтриманні консенсусу та рівномірне розподілення винагороди за створення блоку.

На основі сформованих вимог опишемо акторів системи (таблиця 3.1).

Таблиця 3.1 – Опис акторів

Актор	Короткий опис
Інвестор	Здійснює фінансовий переказ криптовалюти. Здійснює обмін зовнішньої криптовалюти на криптовалюту системи. Відстежує власну історію транзакцій. Відстежує загальний хід транзакцій у системі.
Адміністратор	Отримує детальну статистику платформи. Переглядає інформацію про всіх користувачів у системі. Переглядає інформацію про всі транзакції у системі. Конфігурує глобальні параметри.
Майнер	Бере участь у підтриманні консенсусу системи, отримує винагороду за створення нових блоків

Беручи до уваги додаткові вимоги відносно доступу до внутрішньої функціональності ПС та форм взаємодії з нею, а також з метою формування детальнішого опису вимог, виділимо основні варіанти використання ПС.

Опис ВВ системи наведено у таблиці 3.2.

Узагальнена UML-діаграма варіантів використання наведена на рисунку А.1 (додаток А).

Таблиця 3.2 – Опис варіантів використання

Актор	Найменування ВВ	Опис ВВ
Незареєстрований інвестор	Реєстрація	Створення облікового запису
Зареєстрований інвестор	Авторизація	Авторизація в системі за даними облікового запису
	Перегляд загальної інформації	Перегляд інформації про криптовалюту та хід проведення транзакцій
	Обмін інвестицій на криптовалюту	Відправка коштів на рахунок обмінної системи з отриманням відповідної кількості криптовалюти на власний рахунок
	Переказ криптовалюти	Відправка криптовалюти на рахунок іншого користувача
	Перегляд інформації про особисті транзакції	Можливість переглянути детальні дані про кожну здійснену транзакцію
Адміністратор	Конфігурація загальних параметрів системи	Конфігурація загальних комісій та курсу
	Перегляд детальної статистики	Перегляд детальної інформації про всіх зареєстрованих у системі користувачів та про здійснені ними транзакції
Майнер	Участь у консенсусі	Підтримання консенсусу з отриманням винагороди за створення нових блоків

На основі аналізу предметної області, для якої буде створюватись ПС, сформувано наступні вимоги до самої криптовалюти.

Забезпечення емісії

Емісія – це випуск в обіг нових грошових знаків та платіжних засобів, що викликає збільшення грошової маси. Під час функціонування платіжної системи для кожного нового блоку має генеруватись відповідна кількість віртуальної валюти.

Взаємозамінність валюти

Взаємозамінність валюти полягає у тому, що всі її одиниці є еквівалентними.

Забезпечення обігу. У самій валюті повинні бути реалізовані можливості передачі певної кількості валюти безпосередньо іншому власнику або надання права розпоряджатись цією валютою (Allowance).

Незалежність валюти

Функціонування валюти повинно бути незалежним від будь-яких інших компонентів системи.

Децентралізованість

Після випуску валюти ніхто, включно зі створювачем валюти, не може ніяк впливати на її характеристики, закладені при створенні.

Також було визначено наступні задачі, які має забезпечувати ДПС:

– створення нової віртуальної валюти;

– купівля валюти за інший віртуальний актив через офіційну обмінну платформу;

– обмін валютою між користувачами;

– зручний веб-інтерфейс взаємодії із системою;

– реєстрація та авторизація користувачів;

– взаємодія із системою напряму, без посередників у вигляді веб-інтерфейсу;

– надання інформації про історію транзакцій користувача, загальну історію транзакцій у системі, інформації про криптовалюту (розмір емісії, назва тощо);

– можливість брати участь у підтриманні консенсусу системи з отриманням винагороди за створення нових блоків.

Таким чином, на основі дослідження предметної області криптовалютного інвестування отримано матеріал, в результаті аналізу якого визначено вимоги до ДПС. Наступним етапом буде її проектування.

3.3 Проектування програмної системи

3.3.1 Розробка структури програмної системи

Опишемо моделі потоків даних, що наявні у системі, тобто які дані та в якій послідовності обробляються. Поряд з цим визначимо інформаційні потреби користувачів програмної системи.

Формування дерева зв’язків між функціями залежить від реалізації процесів роботи з БД, блокчейн-частиною та організацією інтерфейсу. Таким чином, ПС можна реалізувати на основі трьох умовно-незалежних програмних модулів.

При роботі з блокчейном користувач посилає через інтерфейс програми (веб-серверну обробку) повідомлення про необхідність виконати деяку дію і передати відповідний результат, представлений на екран. Таким ж способом відбувається виконання операцій над інформацією в БД. Це дозволяє розділити окремі етапи обробки даних та явно вказати модулі ПЗ, що відповідають за відповідні завдання:

а) інтерфейс користувачів:

1) представлення екранних форм для введення вихідних даних;
2) представлення результату обробки даних у зручному для користувачів вигляді;
3) візуалізація інтерфейсу користувача;

б) інтерфейс системи:

1) формування структури і наповнення екранних форм користувача;
2) передача відомостей від користувача на рівень БД;
3) передача відомостей від користувача на блокчейн-рівень;
4) передача користувачеві результату обробки з БД;
5) передача користувачеві результату обробки з блокчейну;
6) специфічна обробка даних без участі блокчейну чи БД;

в) робота з БД:

1) модифікація даних в БД;
2) вибірка даних за інформаційними потребами користувачів;

г) робота з блокчейном:

1) модифікація даних у блокчейні;

2) отримання даних за інформаційними потребами користувачів;

3) обробка даних у мережі блокчейн.

Для такого чіткого поділу функцій між модулями програмного додатку має бути сформована угода форматів повідомлень, які визначають:

а) на рівні БД:

1) отримання повідомлень для виконання процедур чи запитів;

2) повернення результату обробки у вигляді простого повідомлення;

б) на рівні інтерфейсу користувача:

1) отримання сформованого повідомлення для візуалізації у вигляді коду екранної форми, даних для окремих елементів управління екранної форми, коду звітної форми тощо;

в) на рівні блокчейн-частини:

1) отримання блокчейном повідомлень у вигляді виклику функцій розумного контракту для модифікації даних за допомогою протоколу RPC;

2) отримання блокчейном повідомлень у вигляді виклику функцій розумного контракту для читання даних за допомогою протоколу JSON-RPC.

На рівні веб-сервера відбувається відповідна обробка отриманих та переданих даних для подальшого використання у БД, блокчейні чи інтерфейсі користувача.

Процес обробки даних дає можливість виділити різні рівні виконання операцій: внутрішній (рівень роботи БД, рівень роботи з блокчейном); зовнішній (рівень інтерфейсу користувача); проміжний (рівень серверної обробки).

Як було згадано, структуру ПС можна представити наступними компонентами:

– блокчейн-система (розумні контракти);
– оглядач блоків та транзакцій;
– платформа адмністрування;
– обмінна система
– криптогаманець користувача.

Схема взаємодії компонентів системи представлена на рисунку 3.1.

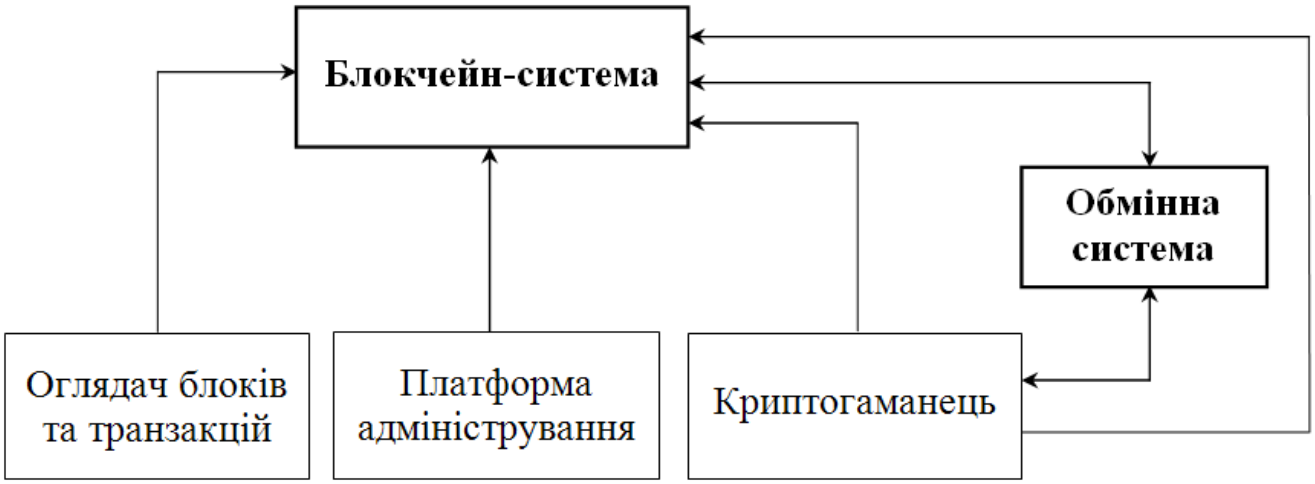


Рисунок 3.1 – Загальна схема взаємодії компонентів системи

Опишемо детальніше кожен із компонентів системи.

Блокчейн-система акумулює основну логіку продажів. Вона є ядром системи та може функціонувати незалежно від інших додатків. Додатки веб-частини взаємодіють з нею через спеціальний протокол віддалених процедур.

Веб-частина складається з наступних компонентів:

– веб-інтерфейс для користувачів;

– веб-інтерфейс для адміністраторів;

– оглядач блоків та транзакцій, доступних всім бажаючим.

Частину даних серверний додаток отримує з блокчейна, частину інформації – із власної БД.

Далі виконаємо декомпозицію модулів, опишемо зв’язки між компонентами та дамо коротке пояснення цих залежностей.

Блокчейн-модуль складається з двох основних розумних контрактів:

– контракт криптовалюти, який відповідає за функціонування валюти;

– контракт продажів, який відповідає за функціонування продажу валюти.

Діаграма взаємодії розумних контрактів зображена на рисунку 3.2.

Для опису декомпозиції веб-додатків розглянемо їхні програмні модулі, а саме, усі пакети, що задіяні у проектуванні з поясненням їх призначення. Структура веб-додатків складається із семи пакетів, кожен з яких представляє певну функціональну особливість.

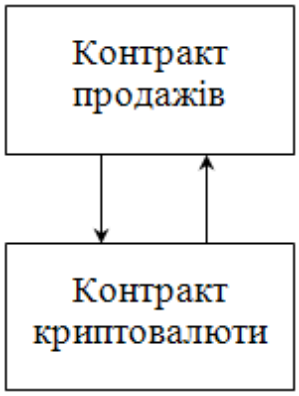


Рисунок 3.2 – Діаграма взаємодії розумних контрактів

Діаграма компонентів користувацьких додатків зображена на рисунку 3.3.

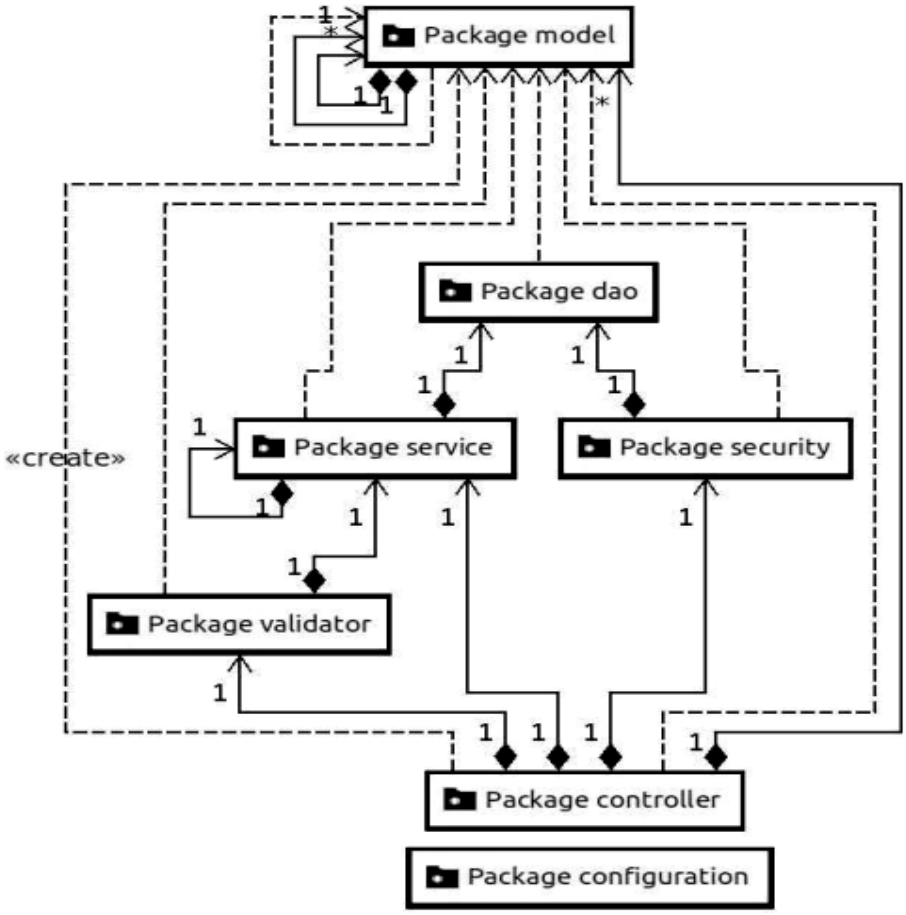


Рисунок 3.3 – Діаграма модулів користувацьких додатків веб-системи

Package model (Пакет моделей) призначений для зберігання класів моделей – структур об’єктно-реляційного відношення. Цей модуль є центральним компонентом шаблону MVC і відображає поведінку додатку, якаа є незалежною від

інтерфейсу користувача. Модель стосується прямого керування даними, логікою та правилами додатку.

Package dao (Пакет об’єктів доступу до БД) надає абстрактний інтерфейс до БД, реалізуючи певні операції без розкриття деталей.

Package service (Пакет сервісів) представляє основну бізнес-логіку системи.

Package controller (Пакет контролерів) описує порядок взаємодії користувача із системою.

Package validator (Пакет валідації) вирішує задачу перевірки вхідних параметрів у системі на коректність з точки зору бізнес-логіки.

Package security (Пакет безпеки) представляє механізми побудови систем аутентифікації та авторизації, а також інші можливості забезпечення безпеки ДПС.

Package configuration (Пакет конфігурації) несе відповідальність за коректне функціонування та контроль процесу роботи системи.

Таким чином, розглянуто структуру основних програмних компонентів системи. Модульна незалежність дозволяє системі залишатися гнучкою за умов зміни вимог до її роботи.

3.3.2 Проектування структури даних

Опишемо детальніше структури даних системи. Дані у системі поділяються на дві групи: інформація, що зберігається у блокчейні, та інформація, що зберігається у БД веб-системи і є необхідною для її функціонування.

Для представлення даних в різних компонентах системи використовуються різні моделі даних з різним призначенням. Якщо блокчейн-частина зберігає масив даних, який критично необхідний для функціонування системи в цілому, то веб-частина містить лише допоміжну службову інформацію, без якої система може функціонувати та яка зберігається лише для забезпечення зручності роботи користувачів (наприклад, історія транзакцій) чи додаткових можливостей.

У блокчейн-додатку зосереджена основна інформація, яку можна логічно розділити на дві частини: контракт цифрової монети та контракт продажів.

Виділимо дані, характерні, для кожної з частин.

Цифрова монета має наступні атрибути:

– назва;

– символ;

– поточний рівень емісії;

– баланси користувачів (значення балансу для кожної адреси).

Контракт продажів має наступні атрибути:

– розмір емісії монет на продаж;

– ціна;

– поточна кількість зібраних інвестицій;

– поточна кількість розданих монет;

– значення отриманих інвестицій від кожного інвестора.

Тепер опишемо об’єкти, які є відображенням таблиць у БД. В системі присутні наступні веб-сервіси: обмінна платформа, криптогаманець, оглядач блоків та транзакцій і платформа адміністрування. Відповідно, кожен із сервісів має свою БД, тому розглянемо структуру кожного з них окремо.

В обмінній системи є користувачі, які можуть реєструватись, вказувати дані облікового запису, а також створювати запити на обмін, які будуть динамічно змінюватись залежно від стадії опрацювання запиту. У БД обмінної системи будуть зберігатись дані про обліковий запис користувача, запити на обмін та дані про згенеровані гаманці для кожного користувача.

Таким чином, структура даних профілю користувача містить такі поля:

– ідентифікатор користувача;

– адреса електронної пошти;

– хеш пароль;

– ім’я;

– прізвище;

– фото;

– адреса криптогаманця;

– адреса гаманця зовнішньої криптовалюти.

Структура даних запиту на обмін:

– обмінна пара;

– очікувана сума транзакції;

– статус;

– курс зовнішньої валюти на момент створення запиту;

– ціна внутрішньої криптовалюти на момент створення запиту;

– адреса криптогаманця відправника;

– ідентифікатор відправника;

– згенерована адреса для прийому коштів;

– кількість криптовалюти яка надійшла;

– ідентифікатор зовнішньої транзакції;

– ідентифікатор внутрішньої транзакції;

– дата, до якої очікується зарахування криптовалюти.

Дані про згенеровані гаманці:

– приватний ключ;

– публічний ключ;

– адреса;

– ідентифікатор запиту.

Структура даних в базі адмінплатформи дещо схожа. В системі присутні користувачі-адміністратори; вони можуть керувати комісією та ціною, а всі їхні дії повинні логуватись і бути видимими всередині платформи. Таким чином, схема структури даних виглядатиме наступним чином.

Структура моделі адміністратора:

– ідентифікатор адміністратора;

– адреса електронної пошти;

– хеш паролю;

– ім’я;

– прізвище;

– фото;

– ідентифікатор адміністратора, який запросив до системи.

Структура даних події (зміни конфігураційних значень) у системі:

– тип події;

– нове значення;

– ідентифікатор адміністратора, який оновив значення.

Оглядач блоків та транзакцій просто зберігає значення про всі перекази та блоки у системі. Ці значення також доступні у самому блокчейні, але отримати їх копію з локальної БД набагато швидше і простіше, тому відбувається кешування.

Дані про переказ:

– адреса відправника;

– адреса отримувача;

– сума переказу;

– сума комісії;

– час здійснення операції;

– ідентифікатор переказу;

– ідентифікатор блоку.

Дані про блок:

– ідентифікатор блоку

– кількість переказів, які вмістив блок;

– час появи блоку;

– висота блоку;

– сума переказів у блоці.

Криптогаманець користувача відображає інформацію про поточний баланс рахунку користувача та його транзакції. Поточний баланс користувача береться напряму з блокчейну, причому, його список транзакцій додатково кешується в БД. Ідентифікація користувача здійснюється шляхом генерування публічної адреси з його секретної фрази, аналогічно до пароля у традиційних системах.

Структура даних транзакцій користувача:

– тип (вхідна чи вихідна);

– адреса відправника

– адреса отримувача;

– сума переказу;

– комісія за переказ;

– ідентифікатор переказу.

Класи, що описують інфологічну модель в ПС, інкапсулюють ядро даних та основний функціонал їхньої обробки і не залежать від процесів введення/виведення даних. Модель структури даних подана на рисунку Б.1 (додаток Б).

3.3.3 Проектування інтерфейсу користувача

Інтерфейс – система правил і засобів, яка регламентує та забезпечує взаємодію декількох процесів або об’єктів. Інтерфейс користувача – система правил і засобів, яка регламентує та забезпечує взаємодію програми з користувачем. Типовий ІК має пристрої введення та виведення. Отримавши від користувача команду, інтерфейс «відповідає» йому, виводячи різного роду інформацію. Представлення інформації зважує на вибір формату, носія, структури, композиції та візуальних прийомів.

Кінцевим результатом розробки ІК повинна бути система, що в зручному вигляді надає користувачам доступ до елементів керування системи та компактно представляє інформацію, отриману від веб-сервера.

Проаналізуємо контингент бажаної аудиторії, яка матиме доступ до сайту. Оскільки користувачами ДПС виступають будь-які люди, які вирішили інвестувати свої кошти, ніяких додаткових вимог до системи не надається. Система має бути зручною та інтуїтивно зрозумілою, забезпечуючи при цьому зрозумілий та швидкий доступ до всіх функцій.

Опишемо загальну структуру компонентів графічного представлення інтерфейсу для кожної веб-платформи.

Платформа обміну має містити наступні сторінки:

– сторінка входу та реєстрації;

– головна сторінка (для створення запиту на обмін та отримання історії запитів);

– сторінка профілю.

Примітка. Проектування інтерфейсу користувача виконано програмним способом за методом еволюційного прототипування.

Розглянемо компоненти інтерфейсу на цих сторінках. Сторінки входу та реєстрації містять лише форми з відповідними полями. Головна сторінка обмінної платформи містить функціонал для інвестування та перегляду історії інвестувань. (рисунок 3.4).

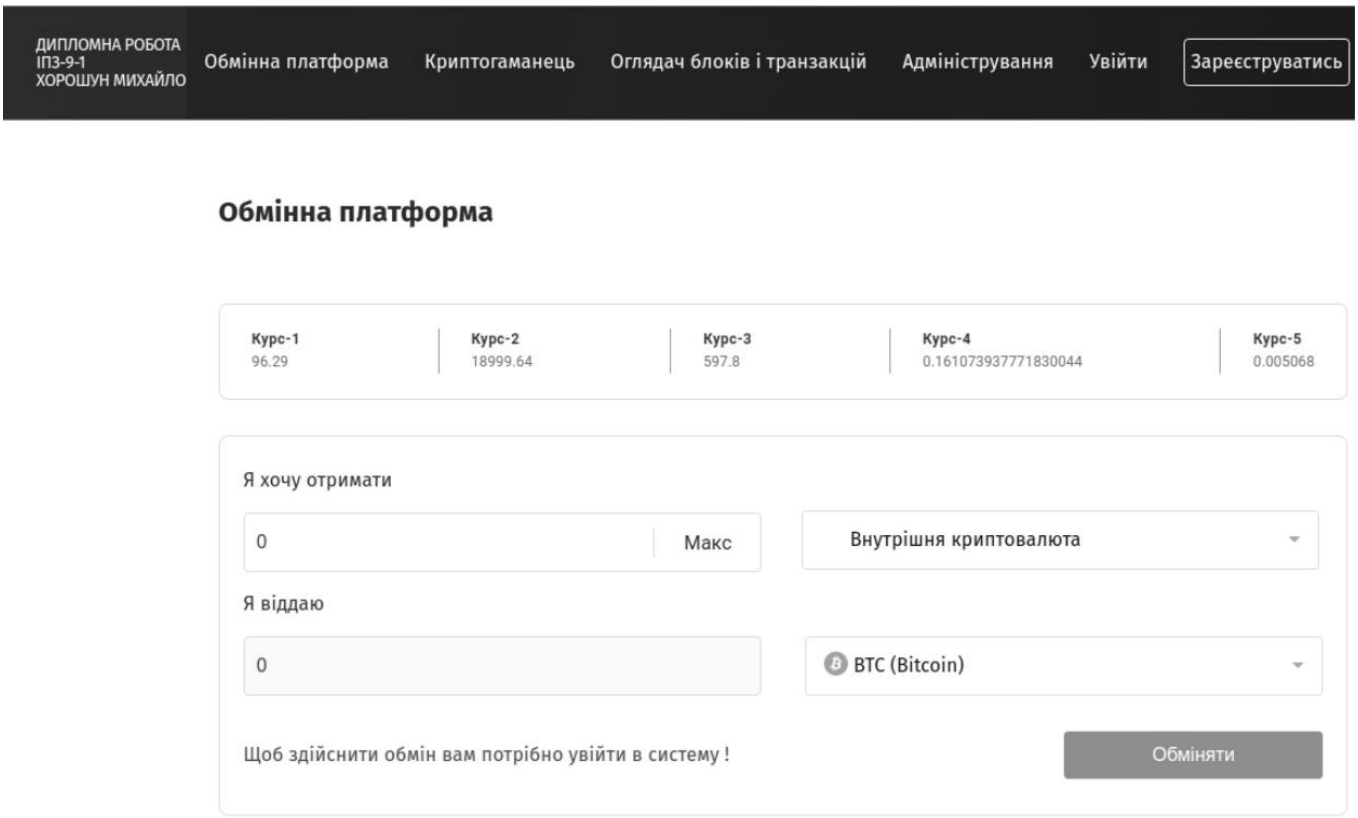


Рисунок 3.4 – Головна сторінка обмінної платформи

Після створення нового запиту на обмін з’явиться інформаційне вікно з таймером (рисунок 3.5). У цьому вікні буде відображена вся необхідна інформація для здійснення обміну.

На головній сторінці обмінної платформи також розміщується вікно історії обмінів; це зображено на рисунку 3.6. Також тут є сторінка користувача, де він може змінювати налаштування свого профілю (рисунок 3.7).

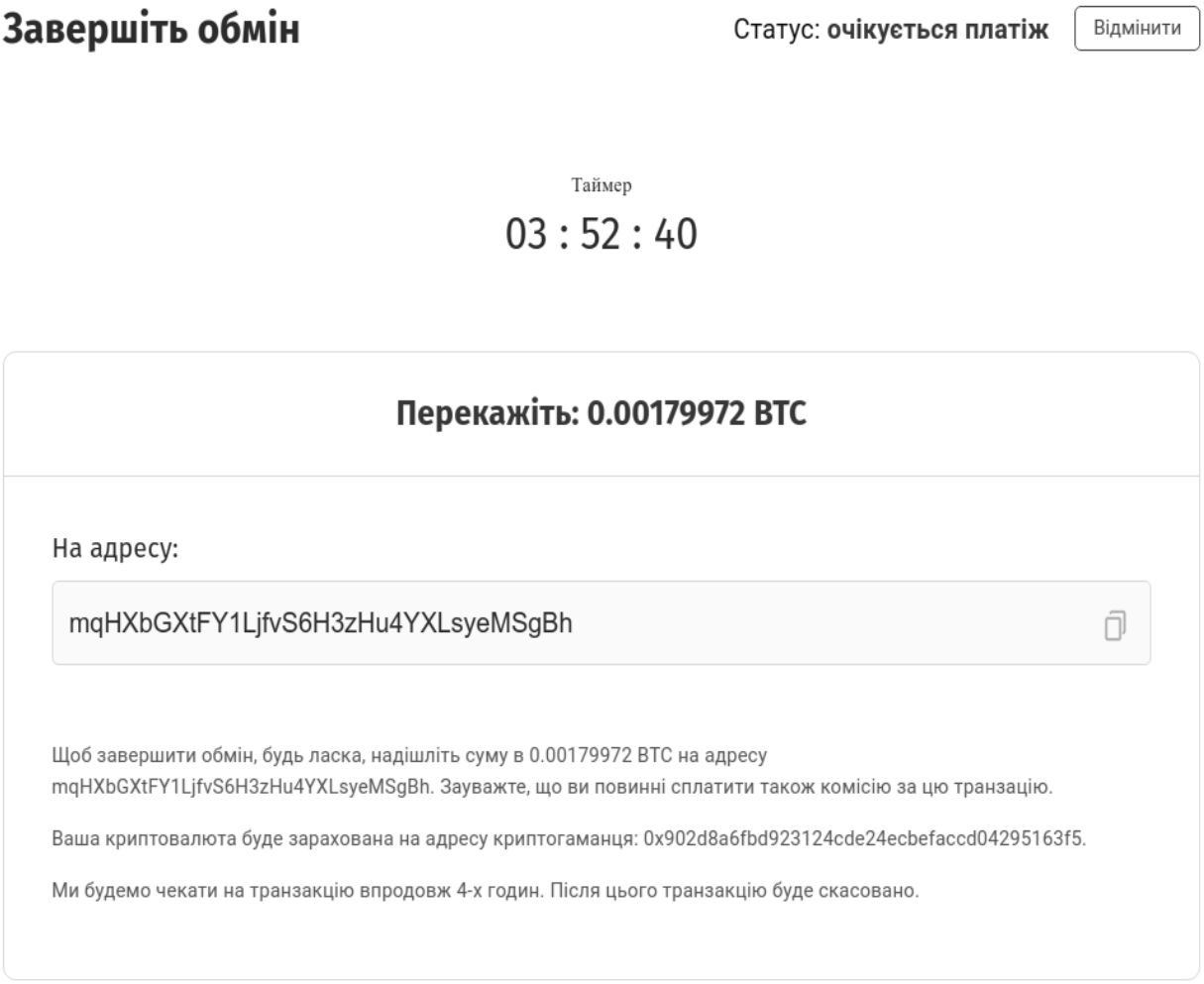


Рисунок 3.5 – Інформаційне вікно обміну

Історія обмінів

Дата	Обмінна пара	Статус	Надіслано	Отримано	Деталі
20.11.2020 01:02	BTC/ІПЗ	Опрацьовано	0.001 BTC	0.526094705882352941 ІПЗ	деталі
20.11.2020 00:23	BTC/ІПЗ	Опрацьовано	0.001 BTC	0.526755 ІПЗ	деталі
19.11.2020 23:46	BTC/ІПЗ	Опрацьовано	0.001 BTC	0.5275114705882352 94 ІПЗ	деталі
02.09.2020 17:38	BTC/ІПЗ	Опрацьовано	0.00012 BTC	0.0399352941176470 58 ІПЗ	деталі

Рисунок 3.6 – Історія обмінів

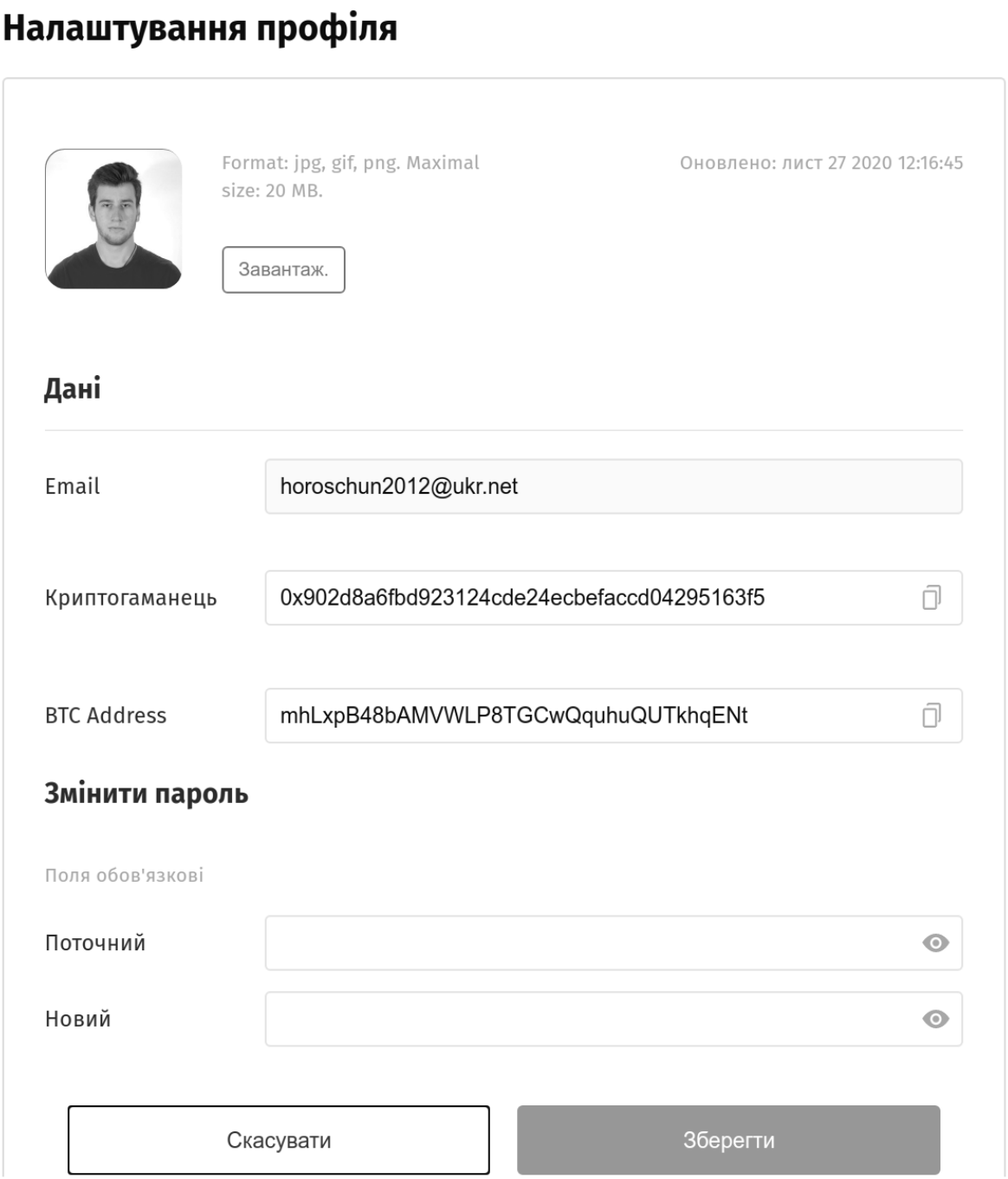


Рисунок 3.7 – Налаштування профілю користувача

Розглянемо інтерфейс криптогаманця – місце, звідки користувач здійснює внутрішні перекази криптовалюти на інші адреси. Інтерфейс історії переказів всередині криптогаманця зображено на рисунку 3.8.

В інтерфейсі криптогаманця також є можливість надіслати свої кошти на іншу адресу, натиснувши на кнопку «Надіслати». При цьому відкриється вікно надсилання переказу (рисунок 3.9).

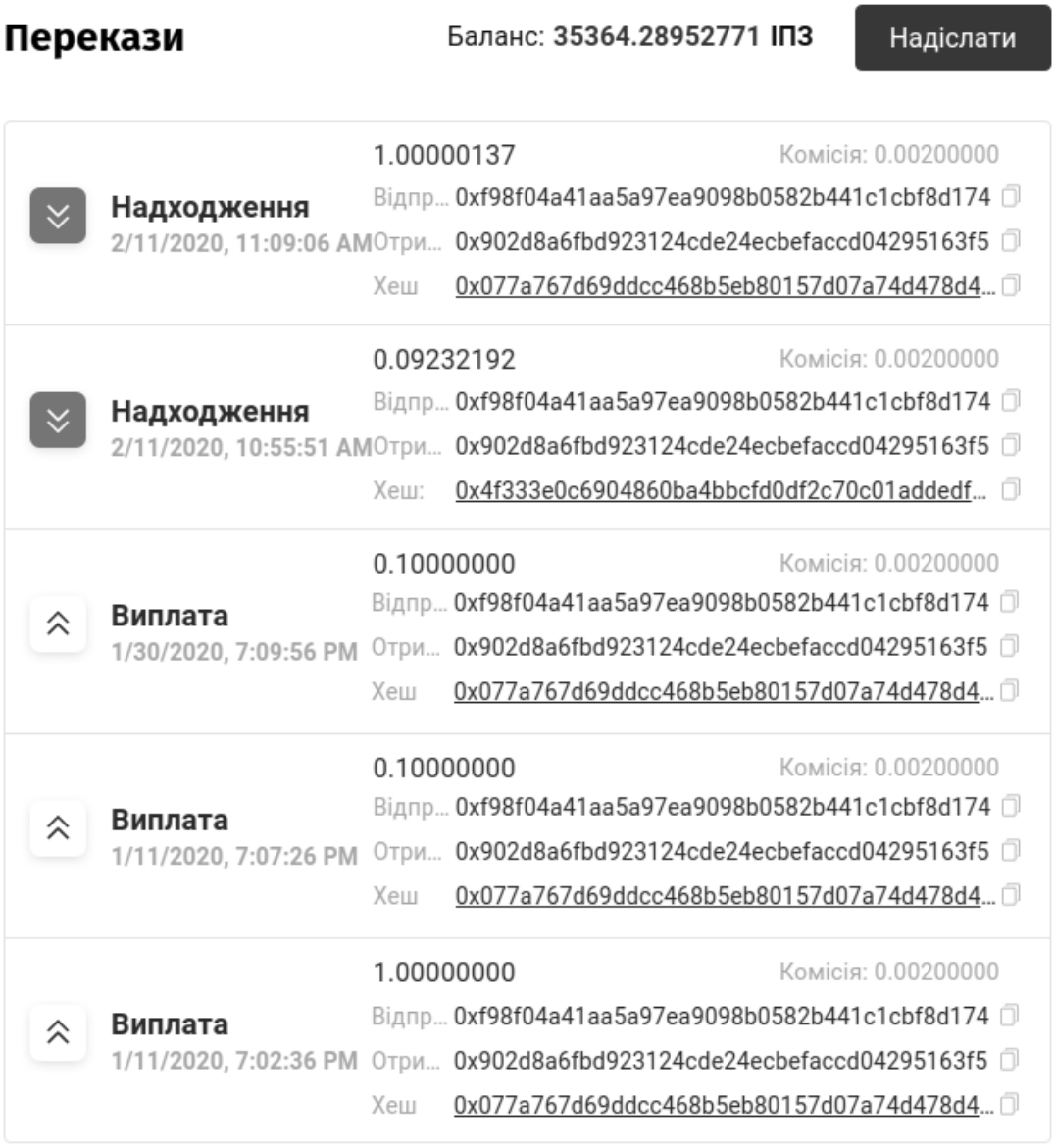


Рисунок 3.8 – Історія переказів всередині криптогаманця

Також розглянемо інтерфейс адміністратора. На платформі адміністрування адміністратор може керувати значеннями комісій для переказів та ціною монети. Для цього призначена сторінка налаштування комісій (рисунок 3.10).

Наостанок розглянемо інтерфейс оглядача блоків та транзакцій. Оглядач блоків та транзакцій – це повністю публічний сервіс, створений для зручного відображення інформації про поточний статус системи, а саме, блоки які створюються, та транзакції в цих блоках. Так, вигляд сторінки з останніми переказами подано на рисунку 3.11, а сторінки з останніми блоками – на рисунку 3.12.

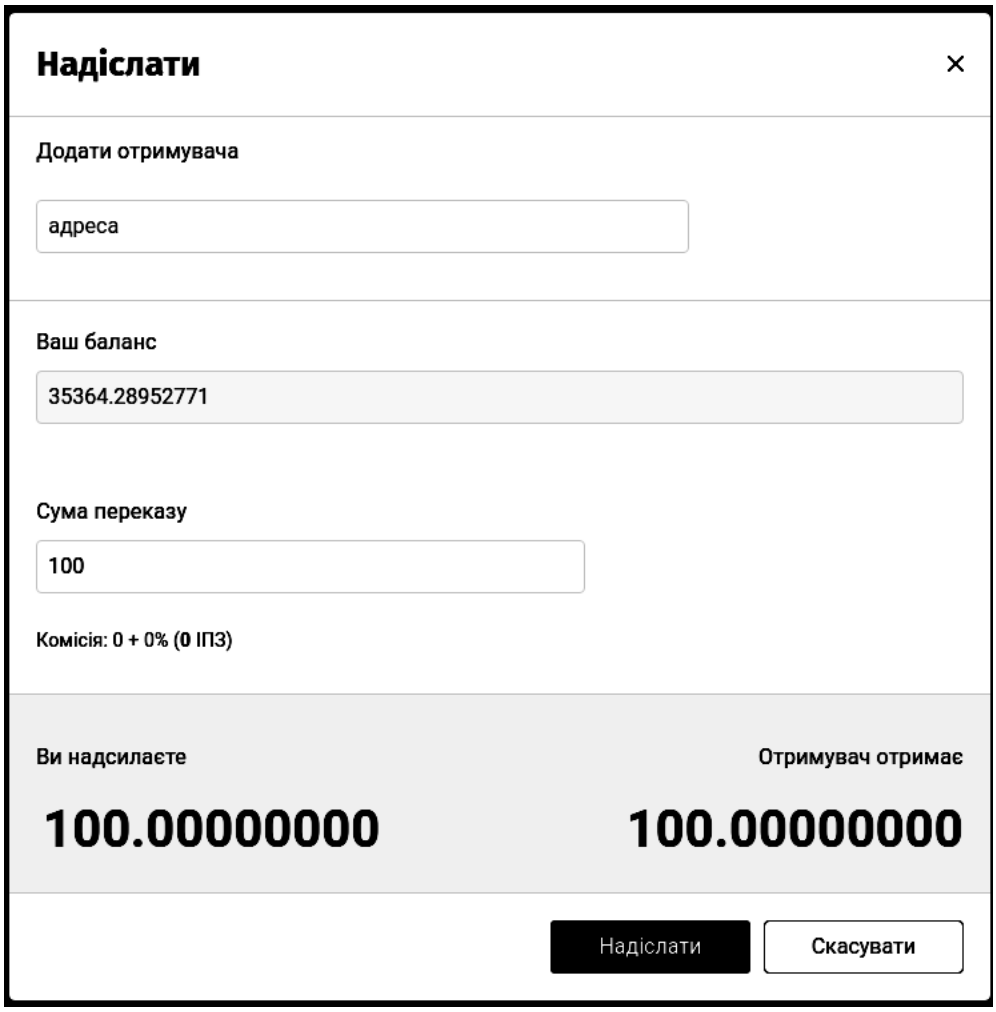


Рисунок 3.9 – Вікно надсилання переказу

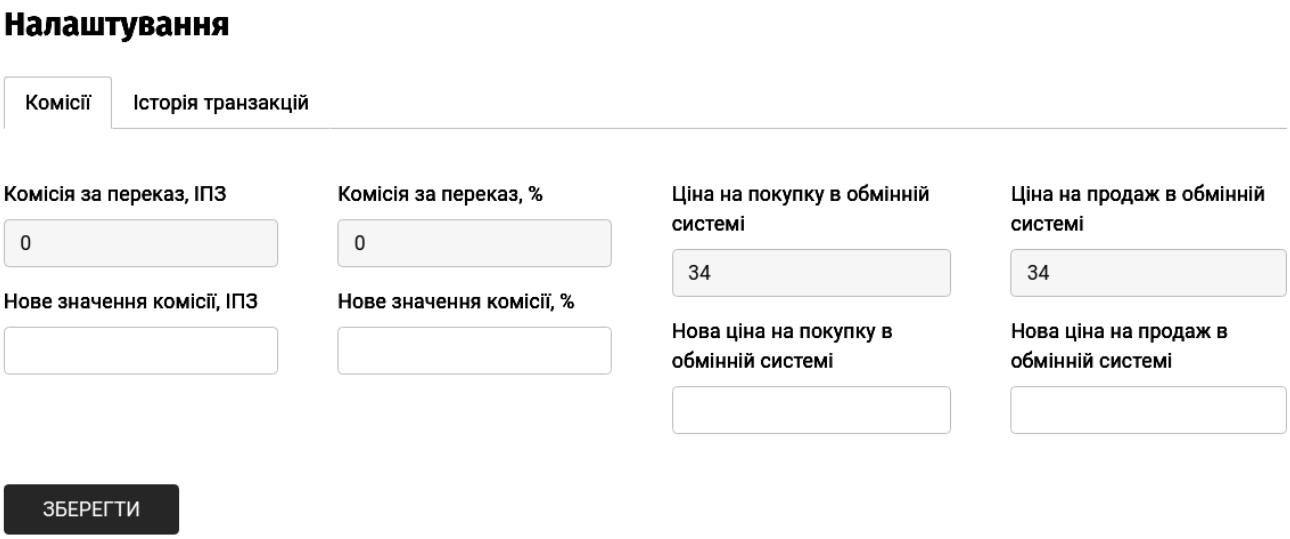
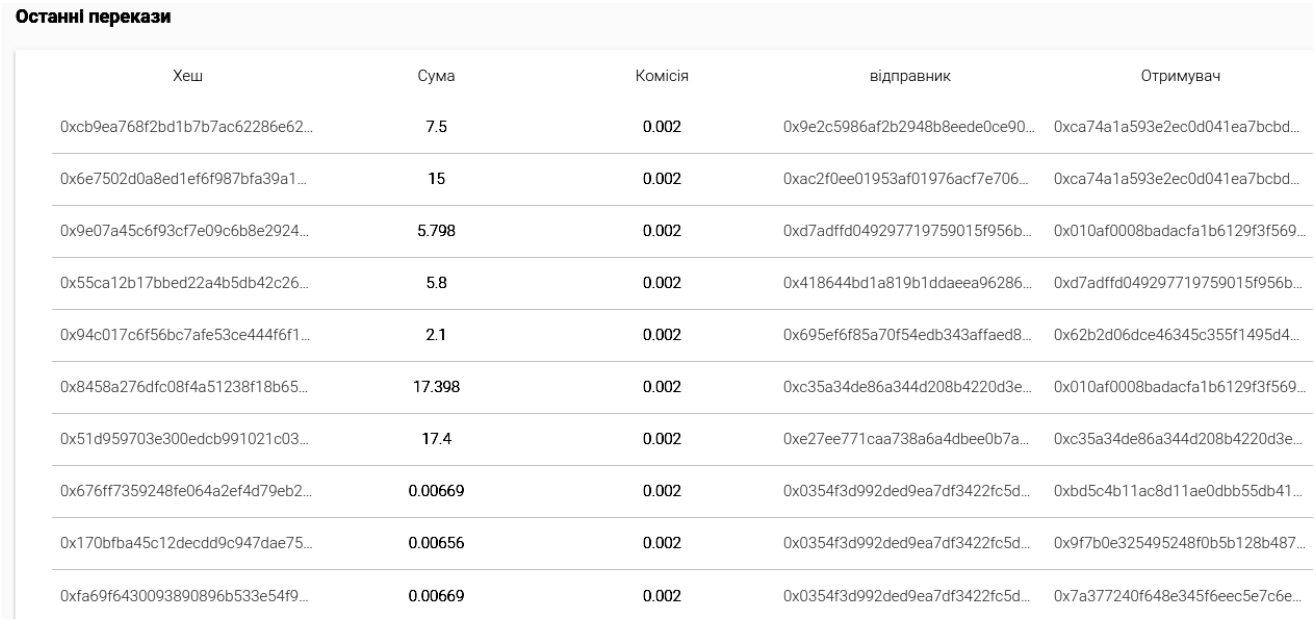


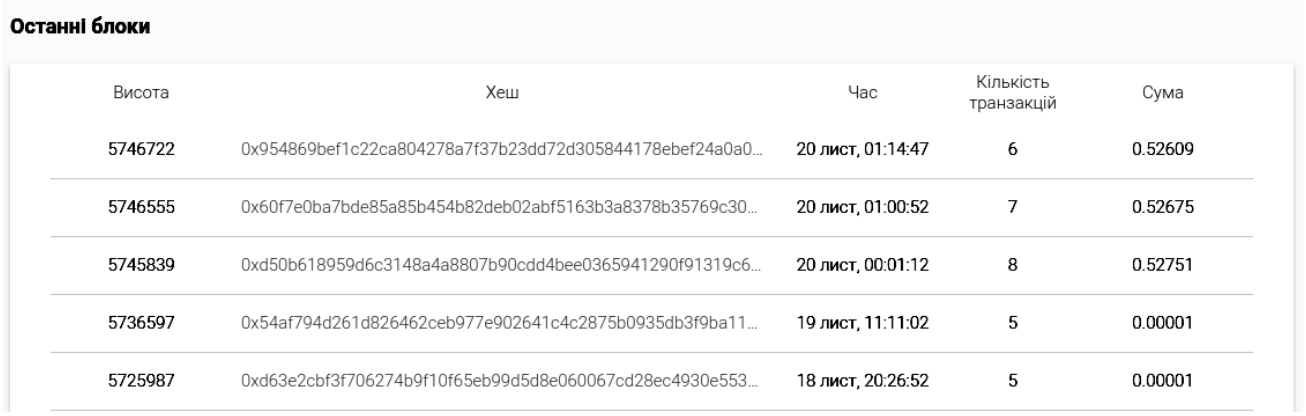
Рисунок 3.10 – Вікно налаштування комісій для адміністратора



Останні перекази

Хеш	Сума	Комісія	відправник	Отримувач
0xcb9ea768f2bd1b7b7ac62286e62...	7.5	0.002	0x9e2c5986af2b2948b8eede0ce90...	0xca74a1a593e2ec0d041ea7bcbd...
0x6e7502d0a8ed1ef6f987bfa39a1...	15	0.002	0xac2f0ee01953af01976acf7e706...	0xca74a1a593e2ec0d041ea7bcbd...
0x9e07a45c6f93cf7e09c6b8e2924...	5.798	0.002	0xd7adffd049297719759015f956b...	0x010af0008badacfa1b6129f3f569...
0x55ca12b17bbed22a4b5db42c26...	5.8	0.002	0x418644bd1a819b1ddaeea96286...	0xd7adffd049297719759015f956b...
0x94c017c6f56bc7afe53ce444f6f1...	2.1	0.002	0x695ef6f85a70f54edb343affaed8...	0x62b2d06dce46345c355f1495d4...
0x8458a276dfc08f4a51238f18b65...	17.398	0.002	0xc35a34de86a344d208b4220d3e...	0x010af0008badacfa1b6129f3f569...
0x51d959703e300edcb991021c03...	17.4	0.002	0xe27ee771caa738a6a4dbee0b7a...	0xc35a34de86a344d208b4220d3e...
0x676ff7359248fe064a2ef4d79eb2...	0.00669	0.002	0x0354f3d992ded9ea7df3422fc5d...	0xbd5c4b11ac8d11ae0dbb55db41...
0x170bfba45c12decdd9c947dae75...	0.00656	0.002	0x0354f3d992ded9ea7df3422fc5d...	0x9f7b0e325495248f0b5b128b487...
0xfa69f6430093890896b533e54f9...	0.00669	0.002	0x0354f3d992ded9ea7df3422fc5d...	0x7a377240f648e345f6eec5e7c6e...

Рисунок 3.11 – Вікно останніх переказів



Останні блоки

Висота	Хеш	Час	Кількість транзакцій	Сума
5746722	0x954869bef1c22ca804278a7f37b23dd72d305844178ebef24a0a0...	20 лист, 01:14:47	6	0.52609
5746555	0x60f7e0ba7bde85a85b454b82deb02abf5163b3a8378b35769c30...	20 лист, 01:00:52	7	0.52675
5745839	0xd50b618959d6c3148a4a8807b90cdd4bee0365941290f91319c6...	20 лист, 00:01:12	8	0.52751
5736597	0x54af794d261d826462ceb977e902641c4c2875b0935db3f9ba11...	19 лист, 11:11:02	5	0.00001
5725987	0xd63e2cbf3f706274b9f10f65eb99d5d8e060067cd28ec4930e553...	18 лист, 20:26:52	5	0.00001

Рисунок 3.12 – Вікно останніх блоків

3.4 Аналіз та вибір засобів реалізації програмної системи

Перед розробниками будь-якої децентралізованої платіжної системи стоїть завдання забезпечити чотири основні вимоги, а саме:

– мережа P2P повинна бути децентралізованою і не контролюватися будь-якими учасниками;

– хакери не повинні мати можливості викрасти конфіденційну інформацію окремих учасників мережі;

– мережа має бути стабільною.

– валюта системи має бути цифровою.

Для створення власних децентралізованих систем зручно використовувати платформи, які вже реалізують ці чотири базові вимоги.

Як було зазначене у першому розділі, найпопулярнішими на сьогодні подібними платформами є Ethereum та Hyperledger. Водночас, Hyperledger не оперує криптовалютами та користується при цьому «каналами» розділення даних, які добре підходять до чітко регульованих галузей (таких як банківська справа чи охорона здоров'я). Ethereum залишається кращим вибором для компаній та розробників. За останні декілька років більшість компаній використовували платформу Ethereum для своїх нових децентралізованих бізнес-моделей.

Враховуючи вищесказане, для розробки доцільно вибрати Ethereum, як блокчейн-платформу.

Як мову програмування розумних контрактів для розробки криптовалюти можна використати Solidity або Serpent. Виходячи з того, що мова програмування Serpent підтримується гірше, ніж Solidity, краще обрати останню.

Обираючи МП веб-додатку, вибір стоїть між такими МП як C++, Java, Javascript та сучасними веб-орієнтованими мовами Ruby або Go.

Обрана МП повинна бути розрахована на завдання, що мають веб-інфраструктуру, використання якої здатно суттєво скоротити час на розробку програми, не змінюючи при цьому її логіку. МП повинна мати низький рівень споживання ресурсів процесора, ефективно використовувати ОЗУ та обчислювальну потужність. Також важливою вимогою до МП є легка інтеграція з блокчейном. Саме такою є мова Javascript, яка задовольняє усім описаним вимогам та має найкращу інтеграцію з блокчейном серед усіх інших мов.

Для клієнтської частини веб-додатку доцільно використати один із найпопулярніших веб-фреймворків javascript – Angular, а для серверної веб- обробки – NodeJS. Для управління базою даних виберемо СКБД MongoDB. Вона легко інтегрується з NodeJS та забезпечує зручний інтерфейс.

Таким чином, було проведено аналіз технологій та сформовано програмний стек, за допомогою якого буде розроблятись програмна система.

3.5 Висновки

Після аналізу предметної області, визначення вимог до ПС та дослідження можливих способів вирішення поставлених задач обґрунтовано проектні рішення, що дають змогу реалізувати вимоги до ПС, забезпечити сумісність та взаємодію різних компонентів ДПС.

У першу чергу було детально розглянуто процедуру обміну зовнішньої криптовалюти на цифрову валюту розроблюваної ДПС. Визначено наступні етапи виконання такого обміну:

а) формування заявки на обмін;

б) підтвердження вхідної транзакції;

в) розрахування комісії та курсу;

г) надсилання криптовалюти на рахунок користувача.

Детально досліджено процедури оптимізації пропускної здатності системи: динамічне збільшення розміру блоку та імплементацію переказу «один до багатьох». Виконання алгоритму динамічного визначення розміру блоку після старту системи передбачається шляхом конфігурування процедури підтримання консенсусу. Алгоритм переказів «один до багатьох» полягає в оптимізації розміру транзакцій шляхом об’єднання декількох переказів користувача в одну транзакцію.

Для підтримання консенсусу в мережі обрано механізм, який не використовує обчислювальну здатність учасників, як параметр підтримання консенсусу, а натомість заснований на необхідності зберігання певної кількості коштів на рахунку для отримання з більшою ймовірністю права для підтвердження чергового блоку.

Також покращено алгоритм розподілу винагороди за створення блоку, зробивши його більш рівномірним. Визначено спосіб, при якому у підсумку всі учасники підтримання консенсусу будуть отримувати відсоток від винагороди після створення кожного блоку.

Після опису алгоритмів, за якими функціонуватимуть ті частини системи, щодо яких розроблялись методи вирішення наявних у галузі проблем, здійснено опис загальних вимог до ПС. На основі сформованих вимог описані актори системи

та виділені основні варіанти використання ПС. Сформовано також вимоги до самої криптовалюти та задач, які повинна забезпечувати ДПС.

На наступному етапі було описано моделі потоків даних, що наявні у системі: які дані та в якій послідовності обробляються. Поряд з цим визначено інформаційні потреби користувачів системи та сформовано дерево зв'язків між функціями роботи з базою даних, блокчейн-частиною та організацією інтерфейсу. Визначено проектну схему реалізації системи на основі трьох умовно-незалежних програмних модулів.

Представлено структуру ПС у наборі компонентів та визначено схему взаємодії між ними. Виконано декомпозицію додатків, описано їхні програмні модулі, а саме усі пакети, що задіяні у проектуванні з поясненням їх призначення.

Розглянуто структуру даних системи. Виділено дві групи даних: інформація, що зберігається у блокчейні, та інформація, яка зберігається у БД веб-системи і необхідна для її функціонування. У блокчейн-додатку зосереджено основну інформацію, яку було логічно розділено на дві частини: контракт цифрової монети та контракт продажів. Виділено дані, характерні для кожної з частин. Після цього описано об’єкти, які є відображенням таблиць у БД. У системі присутні такі веб-сервіси: обмінна платформа, криптогаманець, оглядач блоків та транзакцій і платформа адміністрування. Відповідно, кожен із сервісів має свою БД, тому розглянуто структуру даних кожного сервісу окремо.

Зрештою, проаналізовано таргетингову аудиторію, визначено її функціональні потреби та на цій основі описано загальну структуру компонентів графічного представлення інтерфейсу для кожної веб-платформи. Спроектовано та продемонстровано елементи інтерфейсу для основного функціоналу.

Таким чином, у розділі проведено проектування системи на рівні алгоритмів, архітектури системи, структури даних та інтерфейсу користувача.

Наступним етапом є програмна реалізація отриманих рішень.

4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

4.1 Програмна реалізація

4.1.1 Структура та призначення модулів системи, їхній взаємозв’язок

Програмна система складається з набору наступних компонентів.

Блокчейн-додаток – виконується на блокчейн-платформі і є незалежним від решти компонентів ПС.

Криптогаманець – додаток, що використовує функціонал блокчейн-API для забезпечення інтерфейсу відправки та отримання внутрішніх переказів .

Додаток адміністратора (або кабінет адміністратора) – додаток, що містить основний функціонал для роботи адміністраторів, підтримує зв’язок з БД та зовнішнім хмарним хостингом файлів, використовує функціонал API блокчейн-частини для отримання даних та керування ними.

Обмінна система – додаток, що містить основний функціонал для купівлі криптовалюти за інші цифрові активи, підтримує зв’язок з БД та зовнішнім хмарним хостингом файлів, використовує функціонал API блокчейн-частини для отримання даних та керування ними.

Оглядач блоків та транзакцій – додаток, що використовує функціонал блокчейн-API для забезпечення інтерфейсу відображення інформації про нові блоки та транзакції у системі.

Розглянувши загальну структуру компонентів ПС та взаємозв’язок між її модулями, перейдемо до етапу безпосередньої розробки цих програмних модулів.

Спершу детальніше розглянемо структуру блокчейн-частини системи, оскільки саме вона акумулює основну логіку. Блокчейн-додаток є ядром системи та може функціонувати незалежно від інших додатків. Він складається з двох основних розумних контрактів:

– контракт токена, який відповідає за функціонування монети як валюти;

– контракт продажів, який відповідає за функціонування продажів монет.

Веб-додатки частково використовують API блокчейн-додатку для отримання та керування даними, а також частково використовують власну БД.

Обмінна система містить наступні модулі:

– модуль реєстрації, за допомогою якого відбувається реєстрація акаунтів у системі;

– модуль авторизації, який дозволяє отримати результати перевірки авторизаційних даних користувачів;

– модуль профілю користувача, що дозволяє оновлювати інформацію профілю користувача, а також завантажувати файли фотографій у хмарне сховище;

– модуль опрацювання транзакцій, який містить функціонал по відстеженню покупок криптовалюти та прийманню інших валют як платіжного засобу.

Оглядач блоків містить єдиний модуль для опрацювання транзакцій, який у реальному часі опрацьовує інформацію з блокчейну та оновлює БД, підтримуючи її в актуальному стані.

Платформа адміністрування містить наступні модулі:

– модуль реєстрації, за допомогою якої існуючі адміністратори в системі можуть реєструвати нових;

– модуль авторизації, який дозволяє отримати результати перевірки авторизаційних даних адміністраторів;

– модуль профілю адміністратора, що дозволяє оновлювати інформацію профілю адміністратора, а також завантажувати файли фотографій у хмарне сховище;

– модуль керування системою, який дозволяє адміністраторам встановлювати нові значення ціни монети та комісій на перекази.

Криптогаманець містить наступні модулі:

– модуль для опрацювання у реальному часі інформації з блокчейну про вхідні та вихідні транзакції користувача;

– модуль авторизації користувача за допомогою приватного ключа;

– модуль для формування та надсилання нових запитів на виконання транзакції у мережу.

4.1.2 Розробка програмних модулів

Спочатку необхідно реалізувати «ядро» системи – розумні контракти. Для цього використаємо мову програмування Solidity. Задля зручності розробки також використаємо фреймворк Truffle, що містить базовий функціонал для завантаження контрактів у мережу, компіляції контрактів та їх тестування.

Розпочнемо зі створення контракту валюти. Для цього завантажимо та підключимо бібліотеку open-zeppelin, яка містить базову реалізацію цього стандарту. Створюваний контракт монети має визначити назву, символ, максимальну кількість монет, що може бути випущена, та кількість монет, передбачену для організаторів (включно з їх адресою). Також потрібно створити посилання на контракт продажів для того, щоб перенаправити туди кількість монет, запланованих для продажу. Окрім цього, в контракті монети потрібно реалізувати функцію transferMany для надсилання багатьох переказів одним викликом методу контракту.

Фрагмент контракту криптомонети:

```
pragma solidity 0.5.0;
 contract DiplomaProjectCoin is IDiplomaProjectCoin, StandardToken,
Ownable
{
using SafeMath for uint256;
string public constant name = "Diploma Project Coin"; string public
  constant symbol = "DPC";
uint8 public constant decimals = 18; address public sale;
event Burn(address indexed burner, uint256 value); function
  burnSaleTokens() external onlySale {
uint256 _amount = balances[sale]; balances[sale] = 0;
totalSupply_ = totalSupply_.sub(_amount); emit Burn(sale, _amount);
emit Transfer(sale, address(0), _amount);
}
function transfer(address _to, uint256 _value) public
  spotTransfer(msg.sender, _value) returns (bool) {
return super.transfer(_to, _value);
}
function transferFrom(address _from, address _to, uint256 _value)
  public spotTransfer(_from, _value) returns (bool) {
return super.transferFrom(_from, _to, _value);
}
function transferMany(address[] memory recipients, uint256[] memory
  amounts) public returns (bool) {
```

```
for (uint256 i = 0; i < recipients.length; i++) {
transfer(recipients[i], amounts[i]);
}
return true;
}
}
```

Далі реалізуємо контракт продажів. Контракт продажів має зберігати загальну інформацію про хід продажів та давати змогу здійнити саму покупку за іншу криптоватолюту. Тому всередині контракту користувач має мати можливість дізнатись актуальний курс вхідної криптовалюти на поточний момент часу. Для цього використаємо зовнішній сервіс – oraclize. Цей сервіс дозволяє отримувати дані з-поза блокчейну через технологію снапшотів. Oraclize робить копію веб-сторінки на момент звернення, а потім викликає функцію callback з аргументами результату. Для здійснення покупки користувачу не потрібно викликати будь-яких спеціалізованих функцій. У контракті реалізована спеціальна fallback-функція, яка перейме виконання контракту на себе, якщо на контракт надійшли кошти, і перенаправить інформацію про інвестора та розмір інвестиції для подальшої обробки.

Fallback-функція має вигляд:

```
function () public payable
{
address _sender = msg.sender;
uint256 _funds = msg.value;
uint256 _collector =ETH_COLLECTOR;
     if (!refillers[_sender] && !(owner == _sender))
         _orderCois(_sender, _collector, _funds);
}
```

Далі виконання перенаправляється на функцію orderCoins, у якій виконується перевірка курсу на актуальність. Стандартно, актуальним вважається курс, який було оновлено впродовж минулих 60 хвилин. Якщо курс застарів, то буде здійснено запит до сервісу oraclize, і подальше виконання буде відкладено до моменту, коли надійде відповідь.

Фрагмент функції orderCoins:

```
if (_isRateActual(_collector))
_deliverCoins(_beneficiary, _orderId, _collector, _funds,
collectors[_collector].rate, _currentStage);
else {
orderId = oraclize_query("URL", collectors[_collector] dataSource,
collectors[_collector].gasLimit);
```

Функція deliverCoins відповідає за фактичне зачислення монет для інвестора відповідно до поточного курсу надісланої ним криптовалюти.

Фрагмент функції deliverCoins:

```
uint256 _sum = _funds.mul(_rate).mul(10 ** (COIN_DECIMALS -
collectors[_collector].decimals)).div(10 ** RATE_EXPONENT);
uint256 _usdInvested = _sum.div(10 ** (COIN_DECIMALS - USD_EXPONENT));
uint256 _price = COIN_PRICE[_stage];
uint256 _coins= _sum.mul(10 ** PRICE_EXPONENT).div(_price); if
(distributed.add(_coins) > SALES_SUPPLY)
_coins= SALES_SUPPLY.sub(distributed); collected[_collector] =
collected[_collector].add(_funds);
spent[_beneficiary][_collector] =
spent[_beneficiary][_collector].add(_funds);
raisedUSD = raisedUSD.add(_usdInvested); distributed =
distributed.add(_tokens); coin.transfer(_beneficiary, _tokens);
obtained[_beneficiary] = obtained[_beneficiary].add(_coins);
emit ObtainCOINEvent(_beneficiary, _orderId, _collector, _funds,
_rate, _price, _tokens);
_forwardFunds(_beneficiary, _funds);
```

Для розробки сервісних додатків доцільно скористатись фреймворком, який самостійно згенерує більшу частину коду сервісів та класів-обгорток для моделей. Для зв’язку з БД використаємо клієнт Mongoose. Модульні компоненти сервісних додатків побудовані на основі реактивного фреймворку feathers.js, який автоматично створює для колекцій RestFull-API. Розробнику немає необхідності самостійно реалізовувати передачу даних; фреймворк також піклується про реалізацію сервісів і створенню фільтрів та веб-хуків.

Розглянемо деталі реалізації обмінної системи. Оскільки імплементація більшості модулів є тривіальною, то є сенс зупинитись детальніше на специфічному модулі, який займається фіксуванням зовнішніх транзакцій у мережі, а також модулі взаємодії з розумними контрактами. Для забезпечення взаємодії з контрактами

підключимо спеціальну бібліотеку ethers.js. Реалізація фіксації зовнішніх транзакцій буде залежати від криптовалюти.

Розглянемо приклади для переказів Bitcoin. При створення запиту на обмін для нього генерується унікальна блокчейн-адреса. Ця адреса зберігається протягом дії таймера у спеціальній колекції БД – watchlist. Для фіксування вхідних платежів на адреси з watchlist потрібно створити власний bitcoin-вузол. Підключившись до вузла з додатку, можна відстежувати нові блоки, фільтрувати транзакції за адресою отримувача та зберігати дані про транзакції на рахунки системи.

Фрагмент коду, який розпізнає транзакції Bitcoin на рахунки системи:

```
const zmq = require('zeromq');
const sock = zmq.socket('sub');
const { btcZeroMqTcp } = require("../../config");
module.exports = async function (app) {
await watcher.loadAddressesFromDb(app);
sock.connect(btcZeroMqTcp);
sock.subscribe('rawtx');
let network = bitcoin.networks.bitcoin;
sock.on('message', async function (topic, message) {
if (topic.toString() === 'rawtx') {
var rawTx = message.toString('hex');
var tx = bitcoin.Transaction.fromHex(rawTx);
for (let i = 0; i < tx.outs.length; i++) {
try {
const address = bitcoin.address.fromOutputScript(tx.outs[i].script,
network).toString();

if (watcher.checkAddress(address) == true) {
logger.info("Received BTC tx from " + address + " with value = " +
tx.outs[i].value)
let txHash = tx.getId().toString();
let matchingTxs = await app.service("user-transactions").find({
query: {
tx: txHash
}
});
```

Після розпізнавання транзакції вона очікує підтвердження в три блоки, після чого опрацьовується, і відповідна сума внутрішньої криптовалюти зараховується на рахунок користувача.

Фрагмент коду з перевірки підтвердження транзакції:

```
async function confirmBlockHeight(tx) {
const getData = async url => {
const response = await fetch(url);
// console.log(response)
const json = await response.json();
return json;
};

let txJson = await getData(urlTx);
let txblockHeight = txJson.block_height;
let blockHeight = await getData(urlBlock)
logger.info("GET BLOCK HEIGHT FOR TX =" + txblockHeight + " AND FOR
LAST BLOCK = " + blockHeight + "for tx" + tx);
if (!txblockHeight || !blockHeight) {
return false
}
return parseInt(blockHeight) - parseInt(txblockHeight) > 2

}
```

Фрагмент коду з опрацюваням підтвердженої транзакції:

```
if (txConfirmed) {
let tx = await app.service("user-transactions").get(userTxId);
let coinsWithDecimals= new
Decimal(tx.rate).mul(funds).mul(DECIMALS).div(tx.price).toSignificantD
igits(18).trunc().toString();
let coins = new Decimal(coinsWithDecimals).div(DECIMALS).toString();
let txReceipt = await
app.ethers.market.functions.allocateOrderCoins(txId, tx.beneficiary,
coinsWithDecimals);
let txHash = txReceipt.hash;
await app.service("user-transactions").patch(tx._id, {
coins: coins,
status: "processed",
funds: funds,
eexTx: eexTxHash
})
}
```

Інформація про блоки та транзакції користувача, а також його баланс у системі, реалізовані шляхом виклику АПІ блокчейну.

Фрагмент коду для отримання балансу та транзакцій:

```
export class InvestmentComponent implements OnInit {
private async loadBalance() {
```

```
    const res = await Promise.all([
    this.userStatisticSrv.getBalance(this.accountSrv.account.address),
    this.userStatisticSrv.getAvailableCoins(this.accountSrv.acc
    ount.address)]);
    let b = +res[0];
    this.balance = b ? b.toFixed(4) : 0; b = +res[1];
    this.avlCoins = b ? b.toFixed(4) : 0;
    }
private async loadTransactions() {
    const res = await
    this.userTransactionsService.find(
       {query: {beneficiary: this.accountSrv.account.address}}) as
       Pagination<any>;
    this.rows = res.total ? res.data : [];
    }
```

Для прикладу розглянемо також реалізацію оновлення конфігураційних даних контракту продаж в адміністративній платформі.

Фрагмент коду, що відповідальний за встановлення комісій:

```
if (context.data.fixed && context.data.percentage) {
let newFixedFee = new
Decimal(context.data.fixed).toSignificantDigits(18).trunc().toString()
;
let txReceipt = await context.app.ethers.coin.functions.setFixedFee(""
+ newFixedFee);
let event = {
action: userAction,
value: context.data.fixed,
performedBy: context.params.user.email
}
await context.app.service('events').create(event);
context.result = {
type: context.data.type,
value: context.data.fixed
}
if (parseFloat(context.data.percentage) > 10.) {
throw new Error("Percentage fee should be in range from 0 to 10 %")
}

let newPercentageFee = new
Decimal(context.data.percentage).mul(100).toFixed().toString();
let txReceiptSecond = await
context.app.ethers.coin.functions.setPercentageFee("" +
newPercentageFee);
userAction = "Percentage commission changed.";
let eventSecond = {
action: userAction,
value: context.data.percentage,
```

```
performedBy: context.params.user.email
}
await context.app.service('events').create(eventSecond);
```

Програмний код основних модулів наведено у додатку В.

4.1.3 Реалізація моделі бази даних

Розглянемо реалізацію моделей системи у БД.

В рамках роботи для збереження даних всередині блокчейну напряму використовуються смарт-контракти. Тобто значення будь-яких зовнішніх полів контракту будуть збережені у ньому після завершення транзакції. У зв’язку з цим відпадає необхідність у реалізації додаткової логіки збереження інформації всередині блокчейну. Однак, оскільки виконання операцій збереження інформації у блокчейні працює порівняно повільно, немає сенсу зберігати там дані, які не є критичними для функціонування системи (такі як особисті дані користувачів, адреси зовнішніх гаманців, профілі адміністраторів тощо). Для збереження таких даних доцільно використати звичайну БД. Також отримання даних про транзакції і блоки у блокчейні є досить трудозатратним, особливо, коли потрібно отримати історичні дані. Тому доцільно проводити кешування даних – копіювати дані з блокчейну у локальну БД і при запиті користувачів повертати закешовану версію. Це дасть змогу максимально зменшити як навантаження на систему, так і час її відповіді.

Для веб-частини доцільно використати нереляційну СКБД MongoDB, без чітко визначених залежностей як між моделями, так і всередині самої моделі. Використання NoSQL дозволить спростити механізми кешування, забезпечивши зручний спосіб додавання нових полів до існуючих структур даних при необхідності.

Не потрібно заздалегідь проводити ініціалізацію структур даних, які зберігатимуться у системі; досить лише описати моделі у програмному коді. Програмна бібліотека забезпечить процес об’єктного відображення структури моделі у документі БД.

Розглянемо код моделі користувача:

```
module.exports = function (app) {
const mongooseClient = app.get('mongooseClient');
const users = new mongooseClient.Schema({

email: {type: String, required: true, unique: true},
password: {type: String, required: true},
firstName: {type: String},
lastName: {type: String},
photo: {type: String},
address: {type: String},
btcAddress: {type: String},
ethAddress: {type: String},
}, {
timestamps: true
});

return mongooseClient.model('users', users);
};
```

У цій моделі присутні поля, які також використовуються в інших моделях (так, адреса користувача використовується у моделі транзакції обмінної системи).

Модель транзакції обмінної системи:

```
module.exports = function (app) {
const mongooseClient = app.get('mongooseClient');
const { Schema } = mongooseClient;
const userTransactions = new Schema({
userId: { type: Schema.Types.ObjectId },
pair: { type: String, required: true },
fundsExpected: { type: String, required: true },
status: { type: String, required: true },
rate: { type: String, required: true },
price: { type: String, required: true },
coinsExpected: { type: String, required: true },
funds: { type: String },
tokens: { type: String },
expireAt: { type: Date, default: Date.now() + 4 * 60 * 60 * 1000 },
postProcessingExpireAt : { type: Date, default: Date.now() + 48 * 60 *
60 * 1000 },
tx: { type: String, trim: true},
sender: { type: String },
beneficiary: { type: String },
exchangeAddress: { type: String },
}, {
timestamps: true
});
return mongooseClient.model('userTransactions', userTransactions);
};
```

Однак нам не потрібно встановлювати додаткових зовнішніх ключів. Відображення полів між моделями здійснюється безпосередньо у програмному коді і не існує чітких обмежень на формати та способи зв’язування об’єктів.

Решта моделей у системі створюються аналогічним чином. Зв’язки у структурах полів не визначаються на рівні БД, тому кожна модель на даному етапі реалізації є незалежною і просто характеризує перелік полів об’єкта та їхній тип.

4.1.4 Реалізація методів поліпшення технічних характеристик системи

Опишемо деталі технічної реалізації самої ДПС, які дозволили поліпшити її технічні характеристики.

Раніше було описано процес імлементації обмінної платформи та життєвий цикл транзакції на обмін у цій платформі, а також детально розглянуто імплементацію переказу «один-до-багатьох», який дозволяє проводити декілька переказів всередині однієї транзакції. Розглянемо детальніше також інші реалізації методів поліпшення технічних характеристик ДПС, а саме: рівномірне розподілення винагороди за створення блоку з впровадженням навого алгоритму консенсусу та встановлення динамічного розміру блоку.

Як було згадано раніше, розподілення винагороди за процес майнінгу у традиційній моделі виглядає наступним чином: деяка кількість учасників змагається за право створити блок. У підсумку один з учасників отримує цю можливість, створює блок і отримує відповідний розмір винагороди. Після цього змагання продовжується за новий блок. Більш ніж 90% ДПС використовують, як змагання, обчислення деякої задачі, яка вирішується лише методом «грубої сили» і на вирішення якої йде приблизно передбачувана кількість часу (алгоритм PoW). Таким чином, ймовірність того, що блок, а відповідно і винагорода, дістанеться учаснику з невеликою обчислювальною здатністю порівняно невисока. В результаті користувачу необхідно довгий час підримувати консенсус без винагороди в очікуванні отримання права на створення блоку. Для вирішення цієї проблеми у роботі запропоновано наступні кроки:

– відмовитись від алгоритму консенсусу, що використовує обчислювальну здатність учасників; натомість розглядати, як параметр «ваги», кількість криптовалюти на рахунках учасників (алгоритм PoS);

– проводити емісію монет за допомогою смарт-контракту, який і буде розподіляти винагороду;

– у смарт-контракті розробити функції реєстрації учасників та розподілення винагороди.

Таким чином, алгоритм дій буде наступним:

1 Майнер реєструє свою адресу, як учасник підтримання консенсу, і починає брати участь у створенні нових блоків.

2 Під час створення нового блоку винагорода не потрапляє до одного з учасників, а рівномірно розподіляється між усіма, залежно від їхнього вкладу в консенсус (як параметр, використовується кількість криптовалюти на рахунку).

Фрагмент програмного коду з реалізації логіки розподілення коштів винагороди:

```
// miner addresses
mapping (bytes32 => address) public minersAddress;

event MinerAllocation(address indexed recipient, uint256 indexed
amount);

/**
* Constructor*/
constructor (address _EEXCoin) public {
coin = EEXCoin(_EEXCoin);
}

/**
* @dev Allocate miner coins
*/
function allocateMinerCoins(address _recepient, uint256 _amount)
external onlyOwner() {
uint256 _transferAmount = _amount;
coin.transfer(_recepient, _transferAmount);

emit MinerAllocation(_recepient, _amount);
}
```

Цей механізм розподілення коштів є лише надбудовою над реалізованим в одній з версій Ethereum алгоритмтом PoS. Таким чином, вбудований алгоритм PoS

визначає створювача блоку, а метод смарт-контракту здійснює процес видачі винагороди учасникам.

Розглянемо спосіб реалізації динамічного розміру блоку для транзакцій. Для цього модифікуємо Geth – одного з програмних реалізацій протоколу Ethereum.

Для початку потрібно встановити три вузли. Один з них буде виконувати роль так званого завантажувального вузла (bootnode), два інші, допоміжні, будуть до нього підключатись. Далі необхідно кастомізувати код genesis блоку. Цей блок є першим у системі і в ньому визначається конфігурація поведінки. В нього встановлюється очікувана складність генерування блоків.

Фрагмент конфігурації genesis блоку:

```
{
  "config": {
        "chainId": 0,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
  "alloc"      : {},
  "coinbase"   : "0x0000000000000000000000000000000000000000",
  "difficulty" : "0x20000",
  "extraData"  : "",
  "gasLimit"   : "0x2fefd8",
  "nonce"      : "0x0000000000000042",
  "mixhash"    :
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash" :
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp"  : "0x00"
}
```

Тепер у коди вузлів потрібно додати умову, яка на основі заповненості транзакціями останнього блоку, що був створений у системі, зможе збільшувати та зменшувати розмір наступного блоку.

Фрагмент динамічної зміни розміру блоку:

```
if (lastBlockSize > txsSize){
targetgaslimit = lastBlockSize + lastBlockSize * 0.01
}
```

```
exec("geth --networkid '666' --datadir 'path/to/your/chain/db' --
targetgaslimit " + targetgaslimit + " --rpc --rpccorsdomain
'localhost:8545' --mine", (error, stdout, stderr) => {
if (error) {
console.log(`error: ${error.message}`);
return;
}
if (stderr) {
console.log(`stderr: ${stderr}`);
return; }
});
```

Таким чином, було розглянуто програмну складову запропонованих методів реалізації ДПС, що покликані покращити її базові технічні характеристики.

На наступному етапі розглядатиметься тестування ПС та аналіз ефективності розроблених методів.

4.2 Результати тестування системи та їх аналіз

4.2.1 Вибір методів тестування

Опишемо методи тестування, які будемо використовувати [16].

Модульне тестування (unit testing) – метод тестування ПЗ, який полягає в окремому тестуванні кожного модуля коду програми. Модульні тести, або unit-тести, розробляються у процесі розробки програмістами та, іноді, тестувальниками білої скриньки (white-box testers). Зазвичай, unit-тести застосовують для того, щоб упевнитися, що код відповідає вимогам архітектури та має очікувану поведінку.

Інтеграційне тестування (integration testing) – фаза тестування ПЗ, під час якої окремі модулі програми комбінуються та тестуються разом, у взаємодії. Інтеграційне тестування виконується після модульного тестування та перед верифікацією та валідацією ПЗ. Якщо розглядати цей процес як систему, то на вхід їй подаються модулі, які вже пройшли модульне тестування; далі модулі групуються у більші частини, і виконуються тести, передбачені планом.

Тестування продуктивності (performance testing) – це тестування, яке проводиться з метою визначення, як швидко працює програма або її частина під деяким навантаженням.

Основна частина ДПС працює всередині віртуальної машини Ethereum. Продуктивність розумних контрактів дозволяє обробляти таку кількість запитів та операцій, на яку розрахована потужність мережі. Доцільним методом тестування блокчейн-додатку є модульне тестування, яке дасть змогу перевірити контракти на наявність можливих помилок, а також інтеграційне тестування взаємодії контрактів для виконання усіх можливих завдань.

Оскільки найважливішою ділянкою системи є контракти, їх потрібно перевірити якнайретельніше, забезпечивши повне покриття усіх ділянок коду. Вимоги до веб-частини системи не є такими суворими, оскільки, навіть якщо вона вийде з ладу, система все одно залишиться дієздатною. Однак, щоб цього не трапилось, є сенс провести деякі тестування на продуктивність, оскільки у проекту цілком прогнозовано може з’явитись значна кількість потенційних користувачів. Також потрібно провести модульне тестування веб-системи, щоб забезпечити її нормальне функціонування. Необхідність інтеграційного тестування обґрунтовується наявністю великої кількості модулів і компонентів, які повинні злагоджено працювати.

Для проведення тестування розумних контрактів було використано декілька допоміжних додатків.

Mocha − це фреймворк для тестування, що працює на основі node.js, підтримує роботу з браузерами, асинхронне тестування, а також роботу з будь-якими бібліотеками припущень.

Truffle − фреймфорк для розробки розумних контрактів, що дозволяє автоматизувати процес запуску та виконання тестів.

Ganache − додаток для створення та налаштування локального блокчейну, який дозволяє швидко протестувати зміни стану блокчейна та здійснювати інші операції (як, наприклад, «прокручування» часу у блокчейні вперед, щоб перевірити, як поводитиметься програма у різних часових епізодах).

Ethereum-bridge − програма, яка дозволяє підключатись до oraclize-сервера з локального блокчейна.

Solidity-coverage − програмний пакет для перевірки покриття коду тестами. Принцип його роботи полягає у створенні локальної копії контрактів зі встановленням спеціальних прапорців біля кожного рядка коду.

Для виконання тестування веб-частини додатку використано фреймворки Mocha та Chai (як бібліотеки припущень), фрейморк Jasmine (для автоматизації та надання тестам більшої виразності), Karma (як інструмент моделювання різного оточення для додатків, щоб перевірити їхнє функціонування в різних браузерах).

4.2.2 Розробка тестових сценаріїв

Розпочнемо з тестування блокчейн-частини ДПС. Модульне тестування розумних контрактів полягає у перевірці працездатності окремих методів.

Розглянемо деякі тест-кейси модульного тестування (таблиця 4.1).

Таблиця 4.1 − Тест-кейси модульного тестування розумних контрактів

№	Модуль/ функція	Вихідні дані	Очікуваний результат
1	2	3	4
SC-M-1	Монета/ ініціалізація	Початкові дані ініціалізації: емісія, дані про монету, адреса контракту продажів	Ініціалізація контракту, якщо дані введено коректно. Припинення виконання ініціалізації, якщо в даних є помилки.
SC-M-2	Монета/ переказ монет	Кількість коштів, адреса(и) отримувача(ів)	Переказ коштів на іншу адресу(и), якщо їх достатньо на рахунку. Припинення виконання, якщо коштів недостатньо.
SC-M-3	Продаж/ акредитація інвестора	Адреса інвестора	Акредитація користувача

Кінець таблиці 4.1

1	2	3	4
SC-M-4	Продаж/ оновлення верхньої межі актуальності курсу долара	Кількість годин, протягом яких курс долара вважається актуальним	Встановлює верхню межу, протягом якої курс долара вважається актуальним, якщо ця тривалість менша за 24 години. Інакше: помилка виконання

Інтеграційне тестування розумних контрактів полягає у перевірці працездатності функцій, які вимагають коректної роботи пов’язаних модулів. Для інтеграційного тестування розумних контрактів було розроблено спеціальні setup-сценарії, які розгортали та інтегрували різні частини додатку. Розглянемо деякі тейст-кейси інтеграційного тестування (таблиця 4.2).

Таблиця 4.2 − Тест-кейси інтеграційного тестування розумних контрактів

№	Модуль/ Функція	Вихідні дані	Очікуваний результат
SC-I- 1	Майнинг/ генерування і розподіл винагороди	Список майнерів, розмір винагороди	Контракт розподілення винагороди зараховує новостворені монети на рахунки учасників підтримання консенсусу
SC-I-1	Продаж/ купівля монет	Розмір інвестиції, адреса інвестора	Контракт продажів фіксує покупку і зберігає інформацію про неї, контракт токена збільшує кількість криптовалюти на рахунку інвестора

Модульне тестування веб-частини полягає у перевірці працездатності її методів. Розглянемо деякі тейст-кейси модульного тестування веб-частини (таблиця 4.3).

Інтеграційне тестування роботи веб-частини полягає у перевірці працездатності функцій, які вимагають коректної роботи пов’язаних модулів. Розглянемо деякі тейст-кейси інтеграційного тестування веб-частини (табл. 4.4).

Окрім інтеграційного та модульного методів тестувань, було проведено тестування навантаженням [17]. Для цього був розроблений додатковий клієнт, який постійно викликав процедури ПЗ з метою або викликати збій в її роботі, або змусити систему надто повільно обробляти запити. Емпірично, не вдалось вивести систему з

ладу чи значно сповільнити її роботу при показниках у 1000 запитів за секунду. Найчутливішими є модулі статистики у кабінеті адміністратора, які завантажують велику кількість інформації з БД. Але, оскільки у системі не передбачається наявність надто великої кількості адміністраторів, вимоги до навантажувальної стійкості цих модулів не ставляться.

Таблиця 4.3 − Тест-кейси модульного тестування веб-частини

№	Модуль/ Підмодуль/Функція	Вихідні дані	Очікуваний результат
WA-M-1	Обмінна платформа/ авторизаційний модуль/ аутентифікація	Дані аутентифікації: логін та хеш-пароль	Успішна аутентифікація, якщо дані зареєстровано в системі Повідомлення про помилку, якщо даних не знайдено
WA-M-2	Адмінплатформа/ запрошення нового адміністратора	Електронна пошта нового адміністратора	Успішно реєструється новий акаунт, генерується одноразовий пароль доступу і надсилається на вказану електронну адресу
WA-M-3	Оглядач транзакцій/модуль статистики/останні транзакції	–	Отримання інформації про останні транзакції у мережі

Таблиця 4.4 − Тест-кейси інтеграційного тестування веб-частини

№	Модуль/ Підмодуль/Функція	Вихідні дані	Очікуваний результат
WA-I-1	Криптогаманець/ історія транзакцій користувача	Адреса користувача	Дані про всі транзакції користувача, отримані з БД через API-сервіс, відображаються в екранній формі
WA-I-2	Обмінна платформа/здійснення обміну на іншу криптовалюту	Адреса інвестора, розмір інвестиції	Повідомлення про успішне взяття транзакції в обробку; дані про інвестицію передаються у блокчейн-додаток та дублюються у локальній БД; надсилання монет
WA-I-3	Адмінплатформа/модуль керування даними/ встановлення нової ціни монети	Нова ціна, дані авторизації	Успішно змінена ціна, якщо дані авторизації є коректними; повідомлення про помилку, якщо ні

4.2.3 Аналіз результатів тестування

Тестування ПС здійснювалось у порядку важливості її компонентів. Спершу проводилось тестування розумних контрактів, далі – API та клієнтських додатків. Особлива увага була приділена тому, щоб покрити тести 100% функціоналу розумних контрактів. Результати звіту з покриття блокчейн-додатку тестами представлені на рисунку 4.1.

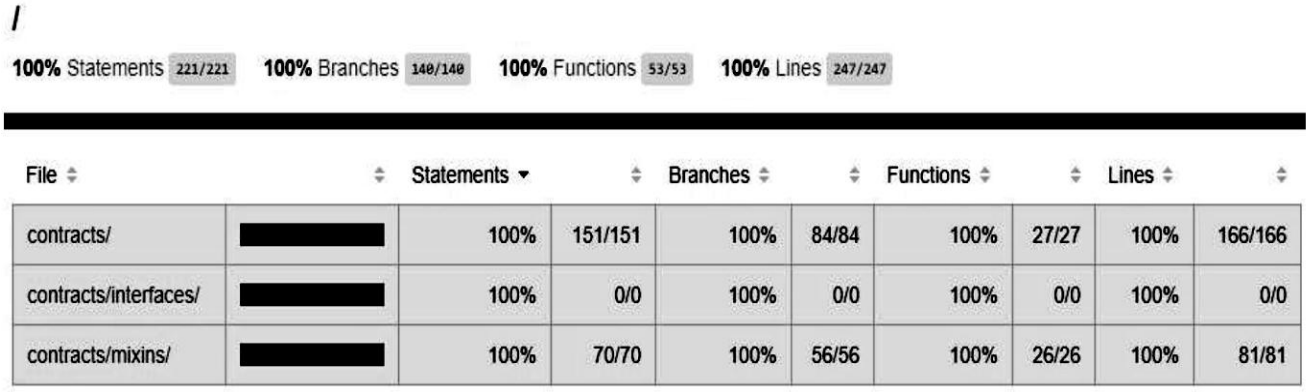


Рисунок 4.1 − Звіт про повне покриття блокчейн-додатку тестами

Було розроблено понад 120 різних тестових сценаріїв з метою забезпечити перевірки всіх можливих розгалужень та некоректних станів. Наведемо для прикладу деякі зі згадних раніше тест-кейсів (таблиця 4.5).

Таблиця 4.5 − Результати перевірки тест-кейсів

№	Опис	Вихідні дані	Очікуваний результат	Різниця між реальним та очікуваним результатом
1	2	3	4	5
SC-M- 1	Ініціалізація монети	Коректний розмір емісії, коректна адреса продажів	Ініціалізація монети	Відсутня
SC-M-2	Монета/ переказ монет	Кількість коштів, адреса(и) отримувача(ів)	Переказ коштів на іншу адресу(и), якщо їх достатньо на рахунку. Припинення виконання, якщо коштів недостатньо.	Відсутня

Продовження таблиці 4.5

1	2	3	4	5
SC-M-3	Продаж/ акредитація інвестора	Адреса інвестора	Акредитація користувача	Відсутня
SC-M-4	Продаж/оновлення верхньої межі актуальності курсу долара	Кількість годин, протягом яких курс долара вважається актуальним	Встановлює верхню межу, протягом якої курс долара вважається актуальним	Відсутня
SC-I- 1	Майнінг/ генерування та розподіл винагороди	Список майнерів, розмір винагороди	Контракт розподілення винагороди зараховує новостворені монети на рахунки учасників підтримання консенсусу	Відсутня
SC-I-1	Продаж/ купівля монет	Розмір інвестиції, адреса інвестора	Контракт продаж фіксує покупку і зберігає інформацію про неї; контракт токена збільшує кількість криптовалюти на рахунку інвестора	Відсутня
WA-M-1	Обмінна платформа/ авторизаційний модуль/ аутентифікація	Дані аутентифікації: логін та хеш-пароль	Успішна аутентифікація, якщо дані зареєстровано у системі; повідомлення про помилку, якщо даних не знайдено	Відсутня
WA-M-2	Адмінплатформа/ запрошення нового адміністратора	Електронна пошта нового адміністратора	Успішно реєструється новий акаунт, генерується одноразовий пароль доступу і надсилається на вказану електронну адресу	Відсутня
WA-M-3	Оглядач транзакцій/модуль статистики/останні транзакції	–	Отримання інформації про останні транзакції у мережі	Відсутня
WA-I-1	Криптогаманець/ історія транзакцій користувача	Адреса користувача	Дані про всі транзакції користувача, отримані з БД через API-сервіс відображаються в екранній формі	Відсутня

Кінець таблиці 4.5

1	2	3	4	5
WA-I-2	Обмінна платформа/ здійснення обміну на іншу криптовалюту	Адреса інвестора, розмір інвестиції	Повідомлення про успішне взяття транзакції в обробку; дані про інвестицію передаються у блокчейн-додаток та дублюються у локальній БД; надсилання монет	Відсутня
WA-I-3	Адмінплатформа/ модуль керування даними/ встановлення нової ціни монети	Нова ціна, дані авторизації	Успішно змінена ціна, якщо дані авторизації є коректними; повідомлення про помилку, якщо ні	Відсутня

Таким чином, на етапі тестування ПС було визначено методики тестування, згідно з якими проведена перевірка якості ПС. На основі отриманих результатів випробувань можна зробити висновок про те, що програмна система відповідає вимогам безпеки, заявлений функціонал працює згідно до вимог.

4.3 Оцінка ефективності моделей та методів вирішення задач

Для отримання інформації про ефективність розробленої ПС варто провести теоретичну та практичну оцінку імплементованих методів.

Розглянемо зміни, які забезпечила розробка наступних компонентів.

1 Власна обмінна платформа. Відсутність необхідності проводити зовнішні платежі на сторонній криптовалютній біржі допомогла підвищити рівень безпеки проведення транзакцій користувачів, а також отримати додаткові нарахування за рахунок комісій.

2 Перехід на алгоритм консенсусу PoS. Відсутність необхідності проводити постійні обчислення значно збільшила енергоефективність системи.

3 Рівномірне розподілення винагороди. Завдяки розробленому алгоритму розподілення монет за створення блоку, винагорода розподіляється рівномірно між

учасниками консенсусу, що забезпечило вищу стабільність отримання своєї частки для усіх майнерів.

4 Розробка динамічного розміру блоку. Завдяки тому, що значення максимального розміру блоку у системі тепер завжди намагається відповідати навантаженню на систему та динамічно зростає або знижується в залежності від навантаження на систему, збільшилась загальна стабільність системи та гарантування проведення транзакцій навіть при збільшенні навантаження.

5 Перекази «один до багатьох». Завдяки функціоналу виклику трансферу криптовалюти, при якому в якості вхідних аргументів метода смарт контракту можна було б передати список адрес отримувачів та необхідні суми переказу, кожному з отримувачів вдалось зменшити навантаження на мережу.

Практична апробація отриманих результатів показала, що завдяки динамічному збільшенню розміру блоку швидкість проходження транзакції тепер не зменшується навіть при зростанні загального навантаження на мережу (кількості транзакцій, що очікують на виконання). Це підтверджується формулою (1) [11], у якій середній час очікування заявки w обернено пропорційний інтенсивності потоку заявок λ, поділеному на кількість транзакцій у блоці. Таким чином, при рівності решти параметрів, якщо з інтенсивністю вхідних заявок співвідносно зростатиме розмір блоку, то загальний час очікування заявки залишатиметься незмінним.

4.4 Висновки

Таким чином, у розділі було детально розглянуто будову компонентів ПС, а саме, блокчейн-додатку, криптогаманця, додатку адміністратора, обмінної системи і оглядача блоків та транзакцій. Склавши загальну структуру компонентів програмної системи та взаємозв’язок між її модулями, перейшли до етапу безпосередньої розробки цих програмних модулів.

Спершу розглянули деталі імплементації блокчейн-частини системи, оскільки саме вона акумулює основну логіку роботи. Реалізували основні смарт-контракти

для функціонування переказів криптовалюти та системи обміну. Далі було розглянуто імплементації веб-додатків, які частково використовують API блокчейн-додатку для отримання та керування даними, а частково використовують власну БД. Тому було реалізовано класи моделей спроектованих сутностей та відображено їх у документах БД.

Також розглянуті процеси реалізації методів поліпшення технічних характеристик системи; проведено детальний огляд алгоритмів та послідовно описано етапи їхньої реалізації.

Наступним кроком стало проведення тестування ДПС. Було описано методи тестування, складено тестові сценарії. На основі результатів можна зробити висновок про те, що ПС відповідає вимогам безпеки, а заявлений функціонал працює у відповідності до вимог.

На завершення було проведено обгрунтування ефективності моделей для методів, реалізованих у ДПС, для підтвердження вирішення поставлених задач.

ВИСНОВКИ

В результаті виконання дипломної роботи було розроблено ДПС для обігу власної криптовалюти.

У першому розділі роботи було досліджено процедури залучення криптовалютних інвестицій та обґрунтувано необхідність розробки системи для автоматизації цієї процедури. При цьому було охарактеризовано структуру предметної області, описано існуючі моделі та підходи для реалізації подібних ПС, виділено їхні плюси та мінуси. На основі проведених досліджень сформовано список проблем, які потрібно вирішити, визначено основні функції створюваної системи, виділено низку функціональних та нефункціональних вимог до ПС, виконано розгорнуту постановку задачі.

У другому розділі досліджено можливі способи вирішення поставлених задач. Моделі та методи вирішення задач у традиційних моделях були удосконалені таким чином, щоб забезпечити вирішення проблем, встановлених на етапі аналізу. Запропоновано оригінальні методи та засоби організації процесів, які б допомогли покращити технічні характеристики існуючих ДПС, підвищити рівень безпеки вкладень користувачів.

У третьому розділі обґрунтовано проектні рішення, що дають змогу реалізувати вимоги описаних алгоритмів, забезпечити сумісність та взаємодію різних компонентів ПС. Розроблено структуру ПС, спроектовано структуру даних та інтерфейсу користувача. Також було обрано засоби, за допомогою яких варто проводити реалізацію ПС, наведено обґрунтування доцільності їх використання.

У четвертому розділі розглянуті питання, що стосуються реалізації ПС на основі розробленого проекту, а також технічні та технологічні характеристики програмної системи, порядок та правила її експлуатації. На основі цих міркувань виконана реалізація системи.

Насамкінець було проведено емпіричне дослідження, спрямоване на доведення працездатності розробленої системи та її функціональної придатності. Зокрема, визначено стратегію тестування, описано та обґрунтувано методи та

методики тестування, сформовано вимоги до проведення експериментів. У підсумку зіставлено і проаналізовано очікувані та фактичні результати тестування.

Розроблений програмний продукт дозволяє автоматизувати процеси залучення інвестицій у технологічні проекти у вигляді емісії та продажу інвесторам нової криптовалюти, а також функціонування криптовалюти як платіжного засобу. У ньому оптимізовано швидкість проходження транзакцій, відсутня необхідністіь залучення третьої сторони (криптовалютних бірж) для купівлі/обміну власної криптовалюти, оптимізувано використання енергозатрат для підтримки функціонування системи шляхом вибору ефективного алгоритму консенсусу. Практична цінність отриманих результатів полягає в успішній розробці моделей та механізмів забезпечення безпечного, прозорого та ефективного процесу створення власної криптовалюти, завдяки поліпшеним характеристикам у порівнянні з традиційними рішенням. Розроблена ПС має високі конкуретні шанси на ринку.

Результати практичної апробації ПС підтверджують її працездатність та відповідність вимогам безпеки. Тому її рекомендовано інтегрувати компаніям, які зацікавлені в притоках інвестиційного капіталу у формі криптовалют.

Таким чином, у результаті виконання дипломної роботи було проведено системний аналіз в галузі криптовалютних інвестицій та на основі отриманих даних спроектовано і втілено у життя концепцію повнофункціональної ДПС для обігу власної криптовалюти.

За результатами дослідження опубліковано дві наукові статті: одна стаття – у фаховому науковому виданні [18] та одна стаття – у збірнику матеріалів Міжнародної наукової конференції [19].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Васильченко З. М. Сучасні тенденції на ринку електронних банківських розрахунків у зарубіжній та вітчизняній практиці / З. М. Васильченко // Банківська справа. – 2017. – № 3. – С. 115–117.

2. Огляд ринку платіжних карток та платіжної інфраструктури України за 2018 рік [Електронний ресурс] // Національний Банк України. – 2019. – Режим доступу до ресурсу: https://bank.gov.ua/file/download?file=Servey_PS-2018.pdf.

3. Олешко А. А. Інноваційні тенденції розвитку безготівкової економіки / А. А. Олешко. // Інвестиції: практика та досвід. – 2018. – No10. – С. 22–25.

4. Момот І. О. Сутність та особливості функціонування криптовалют / І. О. Момот, Ю. Г. Момот, Д. Є. Козенков. // Економіка і суспільство. – 2018. – № 15. – С. 713–719.

5. Гринчук Д. Р. Перспективи впровадження блокчейн-технологій у бізнесі [Електронний ресурс] / Д. Р. Гринчук, М. О. Чупріна // Збірник наукових праць «Сучасні підходи до управління підприємством». – 2019. – Режим доступу: http://spu.fmm.kpi.ua/article/download/180685/180691

6. Золотарьова І. О. Інформаційні технології оптимізації роботи приватного блокчейн за допомогою вибору алгоритму консенсусу / І. О. Золотарьова, Г. О. Плеханова // Системи обробки інформації. – 2020. – № 1. – С. 107–114.

7. Дученко М. М. Вплив криптовалют на економіку країни / М. М. Дученко, Т. В. Павленко // Економіка і суспільство. – 2018. – № 19. – С. 1002–1009.

8. Hileman G. Global cryptocurrency benchmarking study / G. Hileman, M. Rauchs. – Cambridge, 2017. – 114 p.

9. Gervais A. On the Security and Performance of Proof of Work Blockchains / A. Gervais, G. Karame, K. Wüst etc. // CCS'16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. – 2016. – № 10. – P. 3–16.

10. Dorofeyev M. Trends and Prospects for the Development of Blockchain and Cryptocurrencies in the Digital Economy / M. Dorofeyev, M. Ksov, V. Ponkratov // European Research Studies Journal. – 2018. – № 21. – P. 429–445.

11. Иванов Н. Е. Оценка пропускной способности платформы Ethereum на основе математической модели смарт-контракта / Н. Е. Иванов, Р. В. Олейников // Радиотехника. – 2017. – № 191. – С. 40–46.

12. Бойко В. П. Переваги та недоліки використання децентралізованих платіжних систем як інноваційного способу транскордонних розрахунків / В. П. Бойко // Інвестиції: практика та досвід.– 2019.– № 8.– С. 75-82.

13. Стовпова А. С. Порівняльний аналіз та прогноз транзакцій провідних криптовалют // Інвестиції: практика та досвід / А. С. Стовпова. – 2019. – №12. – С. 94-100.

14. Chohan U. Cryptocurrencies: A Brief Thematic Review [Електронний ресурс] / U. Chohan / UNSW Business School. – 2017. – Режим доступу до ресурсу: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3024330.

15. Wohrer M. Design Patterns for Smart Contracts in the Ethereum Ecosystem [Електронний ресурс] / M. Wohrer, U. Zdun // 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). – 2019. – Режим доступу до ресурсу:
https://eprints.cs.univie.ac.at/5665/1/bare_conf.pdf.

16. Куликов С. С. Тестирование программного обеспечения. Базовый курс/ С. С. Куликов. – Минск : Четыре четверти, 2017. – 312 с.

17. Kolawa A. Automated Defect Prevention: Best Practices in Software Management / A. Kolawa, D. Huizinga. / Wiley-IEEE Computer Society Press. – 2007. – 426 p.

18. Хорошун М. Л. Концепції проектування децентралізованої платіжної системи з власною цифровою валютою на базі блокчейн-платформи Ethereum / Г. І. Радельчук, М. Л. Хорошун // Вісник Хмельницького національного університету. Серія «Технічні науки». – 2020. – № 4 (287), Т. 1. – С. 89–93.

19. Хорошун М. Л. Проектування програмної системи для залучення криптовалютних інвестицій на базі блокчейн-платформи ETHEREUM: концептуальні засади / Г. І. Радельчук, М. Л. Хорошун // Materiály XVI Mezinárodní vědecko-praktická konference «Vědecký průmysl evropského kontinentu – 2020», Volume 6 : Praha. Publishing House «Education and Science». – S.75–82.

ДОДАТОК А
(обов’язковий)

ЗАГАЛЬНА ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

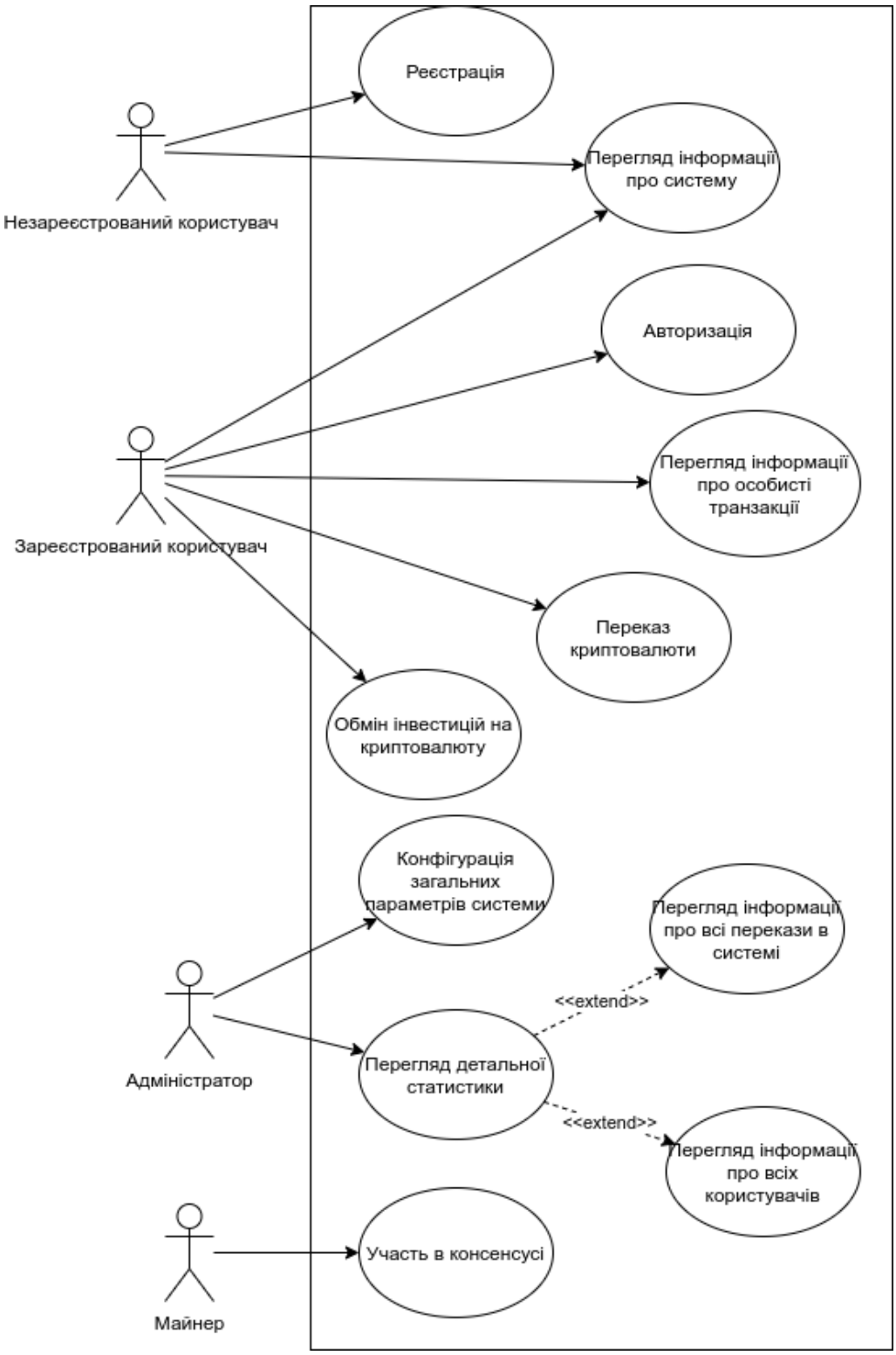


Рисунок А.1 – Діаграма варіантів використання системи

ДОДАТОК Б
(обов’язковий)

МОДЕЛЬ СТРУКТУРИ ДАНИХ

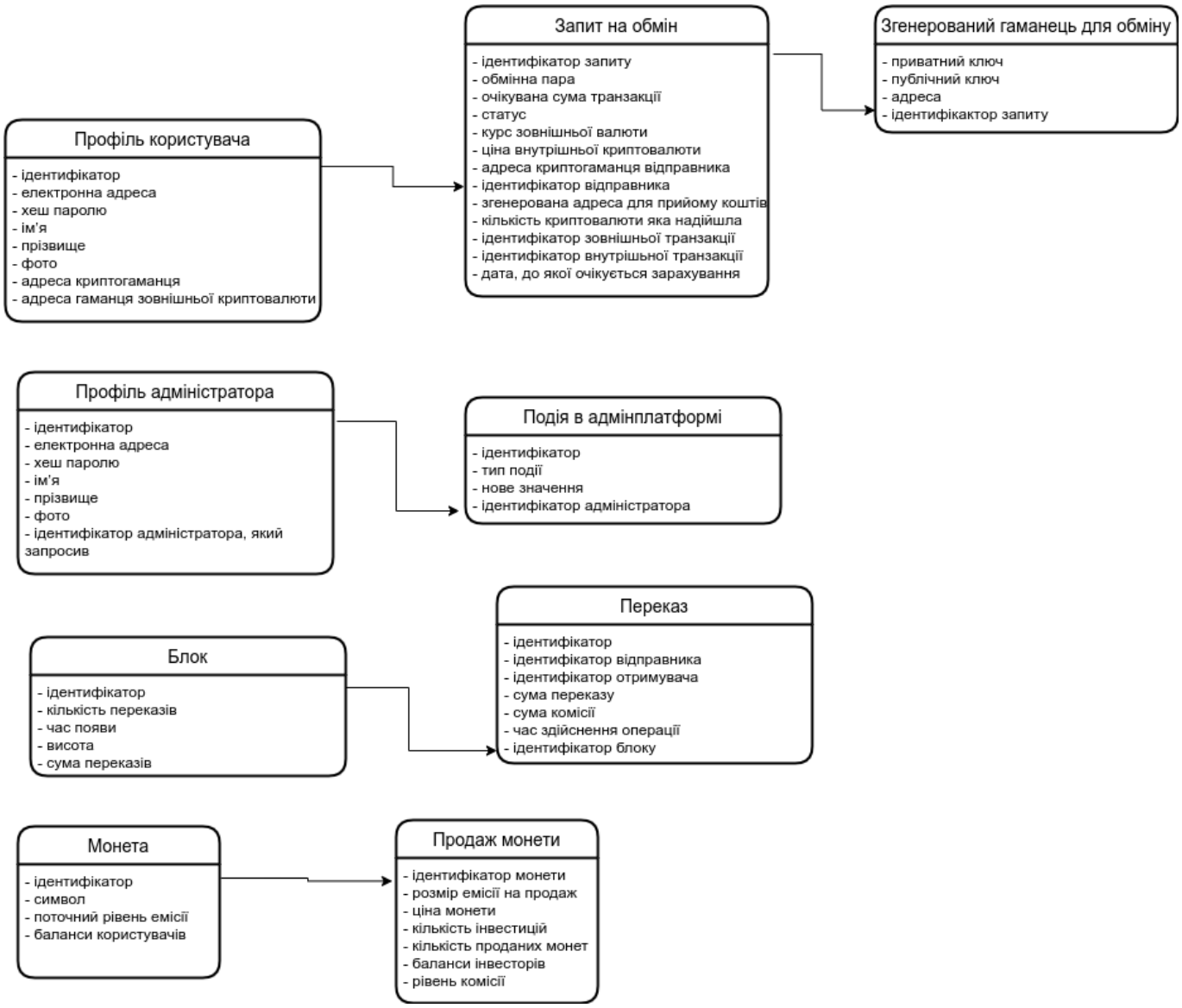


Рисунок Б.1 – Модель структури даних

ДОДАТОК В

(обов’язковий)

ПРОГРАМНИЙ КОД ОСНОВНИХ МОДУЛІВ

В.1 Програмний код модуля смарт-контрактів монети

```
pragma solidity ^0.5.8;
import "openzeppelin-solidity/contracts/coin/Standard/StandardDetailed.sol";
import "openzeppelin-solidity/contracts/ownership/Ownable.sol";
import "openzeppelin-solidity/contracts/math/SafeMath.sol";
import "./mixins/DiplomaConfigurable.sol";
import "./mixins/Standard.sol";
/**
* @title DiplomaCoin
* @dev Diploma Coin implementation
*/
contract DiplomaCoin is Standard, StandardDetailed, DiplomaConfigurable {
using SafeMath for uint256;

// total supply of coins
uint256 constant public MARKET_SUPPLY = 10000000;

// address of DiplomaMarket contract
address public DiplomaMarket;

event TransferWithFee(address indexed who, address indexed to, uint256 amount,
uint256 indexed fee);

/**
* @dev Constructor
*/
constructor () public StandardDetailed("Ethereum Express Coin", "Diploma", 18) {}

/**
* @dev Send all supply of coins to Diploma Market address
* @param _DiplomaMarket DiplomaMarket address
*/
function setDiplomaMarket(address _DiplomaMarket) external onlyOwner() {
require(DiplomaMarket == address(0), "Diploma Market address already exist");
require(_DiplomaMarket != address(0), "Diploma Market address can't be zero
address");

DiplomaMarket = _DiplomaMarket;
feeHolder = _DiplomaMarket;
_mint(DiplomaMarket, MARKET_SUPPLY * (10 ** uint256(decimals())));
emit TransferWithFee(address(0), DiplomaMarket, MARKET_SUPPLY * (10 **
uint256(decimals())), 0);
}

/**
* @dev Standart transfer function with fee deducting from transfer amount
*/
function transfer(address recipient, uint256 amount) public returns (bool) {
address sender = msg.sender;
uint256 netAmount = calcNetAmount(amount);
```

```
uint256 feeAmount = amount.sub(netAmount);
// send coins
_transfer(sender, recipient, netAmount);

// send fee to feeHolder address
_transfer(sender, feeHolder, feeAmount);

emit TransferWithFee(sender, recipient, netAmount, feeAmount);
return true;
}

/**
* @dev Standart transfer from function with fee deducting from transfer amount
*/
function transferFrom(address sender, address recipient, uint256 amount) public
returns (bool) {

uint256 netAmount = calcNetAmount(amount);
uint256 feeAmount = amount.sub(netAmount);

// send coins
_transferFrom(sender, recipient, netAmount);

// send fee to feeHolder address
_transferFrom(sender, feeHolder, feeAmount);

emit TransferWithFee(sender, recipient, netAmount, feeAmount);
return true;
}

/**
* @dev Transfer to many function with fee deducting from transfer amount
*/
function transferMany(address[] memory recipients, uint256[] memory amounts) public
returns (bool) {
for (uint256 i = 0; i < recipients.length; i++) {
transfer(recipients[i], amounts[i]);
}
return true;
}

}

pragma solidity 0.5.8;

import "./DiplomaCoin.sol";
import "openzeppelin-solidity/contracts/ownership/Ownable.sol";
import "openzeppelin-solidity/contracts/math/SafeMath.sol";

contract DiplomaMarket is Ownable {
using SafeMath for uint256;
// Diplomaoin contract
DiplomaCoin public coin;

// used transactions for to prevent spending twice
mapping (bytes32 => bool) public usedTxs;

/**
* @dev Event for purchase coins logging
* @param recipient recipient who obtain coins
* @param amount amount of obtained coins
* @param order order id
*/
```

```
event OrderAllocation(address indexed recipient, uint256 indexed amount, bytes32
indexed order);

event InvestorAllocation(address indexed investor, uint256 indexed amount);
/**
* Constructor
* @param _DiplomaCoin - address of DiplomaCoin contract
*/
constructor (address _DiplomaCoin) public {
coin = DiplomaCoin(_DiplomaCoin);
}

/**
* @dev Allocate investor coins
*/
function allocateInvestorcoins(address _recepient, uint256 _amount) external
onlyOwner() {
uint256 _fee = coin.calcFee(_amount);
uint256 _transferAmount = _amount.add(_fee);
coin.transfer(_recepient, _transferAmount);

emit InvestorAllocation(_recepient, _amount);
}

/**
* @dev Allocate order coins for beneficiary
* @param _order order id
* @param _recepient recepient who has paid coins for coins
* @param _amount coins amount
*/
function allocateOrdercoins(bytes32 _order, address _recepient, uint256 _amount)
external onlyOwner() {
require(!usedTxs[_order], "the order was already processed");
usedTxs[_order] = true;

uint256 _fee = coin.calcFee(_amount);
uint256 _transferAmount = _amount.add(_fee);
coin.transfer(_recepient, _transferAmount);
emit OrderAllocation(_recepient, _amount, _order);
}

}
```

В.2 Програмний код модуля опрацювання запиту на обмін

```
const logger = require('../../logger');
const Decimal = require('decimal.js')
const watcher = require("../../processors/btc-transactions-processor/watcher")
const validateBTCAddress = require('bitcoin-address-validation');
const { iff, isProvider, preventChanges, disallow } = require('feathers-hooks-
common');

module.exports = {
before: {
all: [],
find: [
async context => {
if (context.params.provider) {
responseBody = await
context.app.helpers.authenticate(context.params.headers["authorization"])
if (!responseBody.user) {
throw new Error("Not authenticated: " + responseBody.message)
```

```
}
context.params.query.userId = responseBody.user._id;
console.log(context.params.query.userId)
}
}
],
get: [],
create: [
async context => {
if (context.params.provider && context.params.provider != "server") {
const authToken = context.params.headers["authorization"];
responseBody = await context.app.helpers.authenticate(authToken)
if (!responseBody.user) {
throw new Error("Not authenticated: " + responseBody.message)
}
if (!responseBody.user.diplomaAddress) {
throw new Error("User diploma address is not set")
}
context.data.beneficiary = responseBody.user.diplomaAddress.toLowerCase();

let uncompleteOrders = await context.app.service("user-transactions").find({
query: {
userId: responseBody.user._id,
status: "ordered"
}
});

if (uncompleteOrders.data[0]) {
throw new Error("Previous order is not finished")
}
if (!context.data.pair) {
throw new Error("'Pair' param was not provided");
}

context.data.userId = responseBody.user._id;
context.data.status = "ordered";
if (context.data.pair == "diploma/BTC" || context.data.pair == "diploma/ETH") {
context.data.price = await context.app.helpers.getSaleCoinPrice();
} else {
context.data.price = await context.app.helpers.getCoinPrice();
}
context.data.expireAt = Date.now() + 4 * 60 * 60 * 1000;
switch (context.data.pair) {
case "ETH/diploma":
if (!responseBody.user.ethAddress) {
throw new Error("User eth address is not set")
}
context.data.sender = responseBody.user.ethAddress.toLowerCase();
context.data.rate = await context.app.helpers.getETHRate();
context.data.tokensExpected = new
Decimal(context.data.rate).mul(context.data.fundsExpected).div(context.data.price);
let currentLimit = await context.app.service("user-limits").find({
query: {
userId: context.data.userId
}
})
if (!currentLimit.data[0]) {
currentLimit = await context.app.helpers.getDefaultTokensLimit();
} else {
currentLimit = currentLimit.data[0].limit;
}
logger.info("USER-TRANSACTION:current limit " + currentLimit)
logger.info("USER-TRANSACTION:tokensExpected " + context.data.tokensExpected)
```

```
let limitReached = (parseFloat(context.data.tokensExpected)-2. >
parseFloat(currentLimit))
console.log(limitReached)
if (limitReached == true) {
logger.info("USER-TRANSACTION: limit reacheeed for user " + context.data.userId)
throw new Error("Expected tokens amount bigger than limit amount");
}
break;
case "BTC/diploma":
context.data.rate = await context.app.helpers.getBTCRate();

context.data.exchangeAddress = await context.app.service("btc-exchange-
address").find({ query: { userId: context.data.userId } })
context.data.tokensExpected = new
Decimal(context.data.rate).mul(context.data.fundsExpected).div(context.data.price);
await watcher.addAddress(context.app, context.data.exchangeAddress)
let currLimit = await context.app.service("user-limits").find({
query: {
userId: context.data.userId
}
})
if (!currLimit.data[0]) {
currLimit = await context.app.helpers.getDefaultTokensLimit();
} else {
currLimit = currLimit.data[0].limit;
}
logger.info("USER-TRANSACTION:current limit " + currLimit)
logger.info("USER-TRANSACTION:tokensExpected " + context.data.tokensExpected)

let limReached = (parseFloat(context.data.tokensExpected) > parseFloat(currLimit))
console.log(limReached)
if (limReached == true) {
logger.info("USER-TRANSACTION: limit reacheeed for user " + context.data.userId)
throw new Error("Expected tokens amount bigger than limit amount");
}
break;
case "diploma/BTC":
if (!responseBody.user.btcAddress) {
throw new Error("User btc address is not set")
}

context.data.beneficiary = responseBody.user.btcAddress;
if (validateBTCAddress(context.data.beneficiary) == false) {
console.log(context.data.beneficiary)
throw new Error("Yout BTC address is in invalid format. Please, check your
settings.");
}

context.data.sender = responseBody.user.diplomaAddress.toLowerCase();
context.data.rate = await context.app.helpers.getBTCRate();
context.data.fundsExpected = new
Decimal(context.data.price).div(context.data.rate).mul(context.data.tokensExpected);

if (! await checkIfWalletBalanceEnough(context.app, context.data.fundsExpected)) {
throw new Error("Service is unavailable at the moment. Please, try again later.");
}

let subscriptionData = await
context.app.helpers.createTransferSubscription(context.data.sender);
logger.info("USER-TRANSACTION: created new subscriptionData: " +
JSON.stringify(subscriptionData))
break;
default:
throw new Error("Unsupported exchange pair");
```

```
}
}
}
],
update: [disallow()],
patch: [
async context => {
if (context.params.provider) {
const authToken = context.params.headers["authorization"];
responseBody = await context.app.helpers.authenticate(authToken)
if (!responseBody.user) {
throw new Error("Not authenticated: " + responseBody.message)
}
let orderToPatch = await context.app.service("user-transactions").get(context.id);
console.log(context)
if (responseBody.user._id != orderToPatch.userId) {
throw new Error("UserIDs do not match")
}
if (orderToPatch.status != "ordered" || context.data.status != "canceled") {
throw new Error("Status can't be changed'")
}

preventChanges(
true,
'funds',
'userId',
'pair',
'fundsExpected',
'rate',
'price',
'tokens',
'expireAt',
'tx',
'sender',
'beneficiary',
'exchangeAddress',
'diplomaTx'
)
}
},
],
remove: [disallow()]
},

after: {
all: [],
find: [],
get: [],
create: [
async context => {
const checkOrder = async function (context) {
order = await context.app.service("user-transactions").get(context.result._id);
if (!order.funds)
await context.app.service("user-transactions").patch(context.result._id, { status:
"expired" });
}
setTimeout(await checkOrder, 4 * 60 * 60 * 1000, context);
}
],
update: [],
patch: [],
remove: []
},
```

```
error: {
all: [],
find: [],
get: [],
create: [],
update: [],
patch: [],
remove: []
}
};

async function checkIfWalletBalanceEnough(app, fundsExpected) {
const currentWalletBalance = await app.helpers.getSaleWalletBalance();
logger.info("USER_TRANSACTION: checking current wallet balance = " +
currentWalletBalance);

let fundsExpectedFromOrders = new Decimal(0);
let openOrders = await app.service("user-transactions").find({ query: { status:
"ordered", pair: "diploma/BTC" } });
logger.info("openOrders: " + JSON.stringify(openOrders));
logger.info("I length " + openOrders.data.length)
let i = 0;
for (let i = 0; i < openOrders.data.length; i++) {
logger.info("hu " + openOrders.data[i].fundsExpected);
fundsExpectedFromOrders =
fundsExpectedFromOrders.add(openOrders.data[i].fundsExpected);
}
logger.info("USER_TRANSACTION: checking current open orders funds = " +
fundsExpectedFromOrders);

const finalBalance = new
Decimal(currentWalletBalance).sub(fundsExpectedFromOrders).toString();

if (parseFloat(fundsExpected) > parseFloat(finalBalance)) {
logger.info("USER_TRANSACTION: not enough funds(balance = " + finalBalance + " and
fundsExpected = " + fundsExpected);
return false;
}
logger.info("USER_TRANSACTION: enough funds(balance = " + finalBalance + " and
fundsExpected = " + fundsExpected);
return true;
}
```

В.3 Програмний код модуля встановлення комісій адмінплатформи

```
const { authenticate } = require('@feathersjs/authentication').hooks;
const { disallow } = require('feathers-hooks-common');
const logger = require('log4js').getLogger();
const { mnemonic, coinAddress, marketAddress, diplomaNetworkUrl } =
require('../../config');
const Decimal = require('decimal.js')

module.exports = {
before: {
all: [],
find: [
async context => {
let contractFixedFee = await context.app.ethers.coin.functions.fixedFee();
let fixedFee = "" + parseInt(contractFixedFee._hex, 16) / 10 ** 18;
let contractPercentageFee = await context.app.ethers.coin.functions.percentageFee();
let percentageFee = "" + parseInt(contractPercentageFee._hex, 16) / 10 ** 2;
```

```
switch (context.params.query.type) {
case "fixed":
context.result = fixedFee;
break;
case "percentage":
context.result = percentageFee;
break;
default:
context.result = { fixed: fixedFee, percentage: percentageFee }
break;
}
}
],
get: [disallow()],
create: [disallow()],
update: [disallow()],
patch: [
authenticate('jwt'),
async context => {
let userAction;
if (context.data.fixed && context.data.percentage) {
if (parseFloat(context.data.fixed) < 0 || parseFloat(context.data.fixed) > 0.5) {
throw new Error("Fixed fee should be in range from 0 to 0.5 diploma")
}
let txReceipt = await context.app.ethers.coin.functions.setFixedFee("" +
newFixedFee);
logger.info("TX RECEPT: " + txReceipt)
userAction = "Fixed commission changed";
let event = {
action: userAction,
value: context.data.fixed,
performedBy: context.params.user.email
}
await context.app.service('events').create(event);
context.result = {
type: context.data.type,
value: context.data.fixed
}
if (parseFloat(context.data.percentage) > 10.) {
throw new Error("Percentage fee should be in range from 0 to 10 %")
}
let newPercentageFee = new
Decimal(context.data.percentage).mul(100).toFixed().toString();
let txReceiptSecond = await context.app.ethers.coin.functions.setPercentageFee("" +
newPercentageFee);
userAction = "Percentage commission changed.";
let eventSecond = {
action: userAction,
value: context.data.percentage,
performedBy: context.params.user.email
}
await context.app.service('events').create(eventSecond);

let emailEvent = {
action: "Percentage commission changed. Fixed commission changed.",
value: "fixed: " + context.data.fixed + " percentage: " + context.data.percentage,
performedBy: context.params.user.email
}
await context.app.emails.sendEventForAllAdmins(emailEvent)
context.result = {
fixed: context.data.fixed,
percentage: context.data.percentage
}
} else if (context.data.fixed) {
```

```
if (parseFloat(context.data.fixed) < 0 || parseFloat(context.data.fixed) > 0.5) {
throw new Error("Fixed fee should be in range from 0 to 0.5 diploma")
}
let txReceipt = await context.app.ethers.coin.functions.setFixedFee("" +
newFixedFee);
logger.info("TX RECEPT: " + txReceipt)
userAction = "Fixed commission changed";
let event = {
action: userAction,
value: context.data.fixed,
performedBy: context.params.user.email
}
await context.app.service('events').create(event);
await context.app.emails.sendEventForAllAdmins(event)
context.result = {
fixed: context.data.fixed
}
}
else if (context.data.percentage) {
if (parseFloat(context.data.percentage) > 10.) {
throw new Error("Percentage fee should be in range from 0 to 10 %")
}
let newPercentageFee = new
Decimal(context.data.percentage).mul(100).toFixed().toString();
let txReceipt = await context.app.ethers.coin.functions.setPercentageFee("" +
newPercentageFee);
logger.info("TX RECEPT: " + txReceipt)
userAction = "Percentage commission changed";
let event = {
action: userAction,
value: context.data.percentage,
performedBy: context.params.user.email
}
await context.app.service('events').create(event);
await context.app.emails.sendEventForAllAdmins(event)
context.result = {
percentage: context.data.percentage
}
} else {
throw new Error("Bad request");
}

}
],
remove: []
},
};
```

В.4 Програмний код модуля налаштувань профілю користувача

```
const { iff, isProvider, preventChanges, disallow } = require('feathers-hooks-
common');
const { restrictToAdmin } = require('../../hooks/authentication');
const { restrictToRoles } = require('feathers-authentication-hooks');
const { authenticate } = require('@feathersjs/authentication').hooks;
const { addVerification, removeVerification } = require('feathers-authentication-
management').hooks;
const { sendVerificationEmail } = require('../../hooks/verification');
const logger = require('log4js').getLogger();
const hooks = require('feathers-hooks-common');
const bcrypt = require('bcryptjs');
const {
```

```
hashPassword, protect
} = require('@feathersjs/authentication-local').hooks;

const restrict = [
authenticate('jwt'),
iff(isProvider('external'),
restrictToRoles({
roles: ['admin', 'super-admin'],
idField: '_id',
ownerField: '_id',
owner: true
}))
];

module.exports = {
before: {
all: [],
find: [
async context => {
if (Object.entries(context.params.query).length === 0 &&
context.params.query.constructor === Object) {
throw new Error("Please, provide email address")
}
return context;
}
],
get: [...restrict],
create: [
async context => {
const duplicates = await context.app.service('users').find({
query: {
email: context.data.email
}
});
if (duplicates.data.length) {
throw new Error("Email is already registered")
}return context;
},
hashPassword('password'),
addVerification()
],
update: [disallow()],
patch: [
authenticate('jwt'),
async context => {
if (!context.id) {
context.params.query = { _id: context.params.user._id }
}
if (context.data.diplomaAddress == "") {
context.data.diplomaAddress = undefined;
}
if (context.data.ethAddress == "") {
context.data.ethAddress = undefined;
}
if (context.data.diplomaAddress == "") {
context.data.diplomaAddress = undefined;
}

if (context.data.password) {
if (context.params.user) {
if (context.data.password.new == "") {
context.data.password = context.params.user.password
} else {
```

```
if (await bcrypt.compare(context.data.password.current, context.params.user.password)
== false) {
throw new Error("Current password is incorrect.");
}
context.data.password = await bcrypt.hashSync(context.data.password.new);
}
}
}

if (context.data.diplomaAddress) {
const duplicates = await context.app.service('users').find({
query: {
diplomaAddress: context.data.diplomaAddress.toLowerCase()
}
});
if (duplicates.data.length && duplicates.data[0].email != context.params.user.email)
{
throw new Error("diploma address is already registered")
}
context.data.diplomaAddress = context.data.diplomaAddress.toLowerCase()
}
if (context.data.ethAddress) {
const duplicates = await context.app.service('users').find({
query: {
ethAddress: context.data.ethAddress.toLowerCase()
}
});
if (duplicates.data.length && duplicates.data[0].email != context.params.user.email)
{
throw new Error("ETH address is already registered")
}
context.data.ethAddress = context.data.ethAddress.toLowerCase();
}
if (context.data.btcAddress) {
const duplicates = await context.app.service('users').find({
query: {
btcAddress: context.data.btcAddress
}
});
if (duplicates.data.length && duplicates.data[0].email != context.params.user.email)
{
throw new Error("BTC address is already registered")
}
}
```

ДОДАТОК Г

(обов’язковий)

КОРОТКА ІНСТРУКЦІЯ ДЛЯ КОРИСТУВАЧІВ

Розглянемо порядок дій для інвестора, який має намір здійснити крипто-валютну інвестицію.

Для початку користувачу потрібно зареєструвати новий обліковий запис (якщо його ще не існує). Після входу на сайт перед ним з’явиться вікно з пропозицією зареєструватись (рисунок Г.1), де йому потрібно ввести адресу своєї електронної пошти і пароль.

Рисунок Г.1 – Вікно реєстрації у системі

Після введення усіх необхідних даних користувач вважається зареєстрованим і може увійти в систему, однак для безпеки акаунта варто також підтвердити електронну пошту, перейшовши за лінком в отриманому після реєстрації листі.

При наступному вході у систему можна натискати посилання «У мене є акаунт», який переведе користувача на сторінку входу (рисунок Г.2).

Після введення авторизаційних даних на сторінці входу користувач потрапляє до головної сторінки обмінної платформи, яка зображена на рисунку Г.3.

На цій сторінці користувач може ознайомитись з актуальними курсами криптовалют для обмін та створити власну заявку. Для створення заявки користувач повинен заповнити поля «Я хочу отримати» або «Я віддаю», а також вибрати з випадаючого списку правильний варіант обмінної пари і після цього натиснути

«Обміняти». Але сама процедура обміну поки недоступна, оскільки користувач ще не вказав свій криптогаманець і тому система не знає, куди йому відправляти кошти.

Тому наступним кроком є створення криптогаманця. Для цього слід перейти до пункту «Криптогаманець» з верхнього меню (рисунок Г.4).

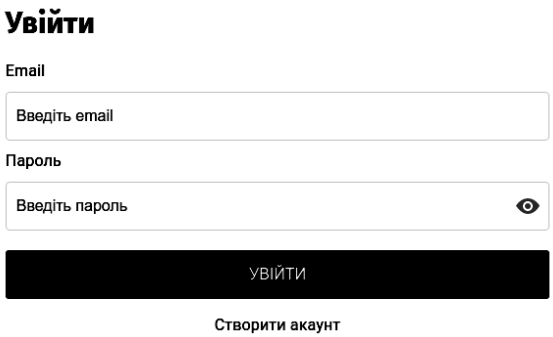


Рисунок Г.2 – Вікно входу у систему

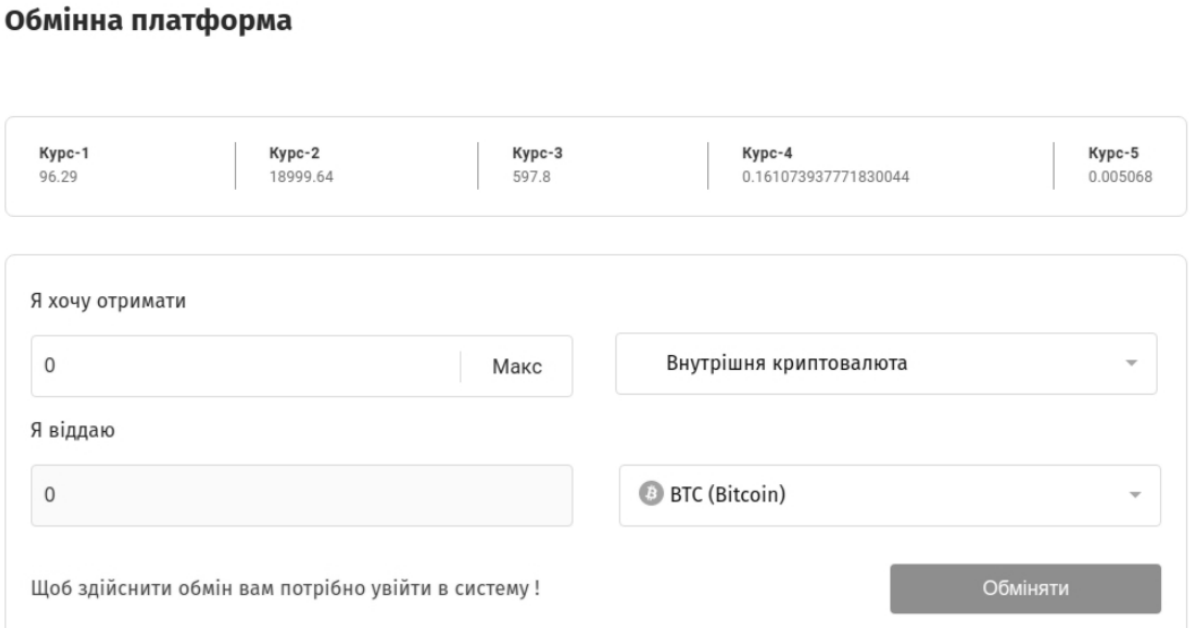


Рисунок Г.3 – Вікно обмінної платформи

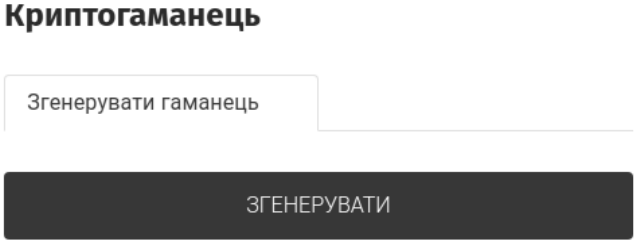


Рисунок Г.4 – Вікно генерування криптогаманця

Після натискання на кнопку генерування користувачу стає доступною публічна адреса гаманця та ключ доступу до нього. Ці дані потрібно зберегти. Далі варто перейти на сторінку налаштувань, яка зображена на рисунку Г.5.

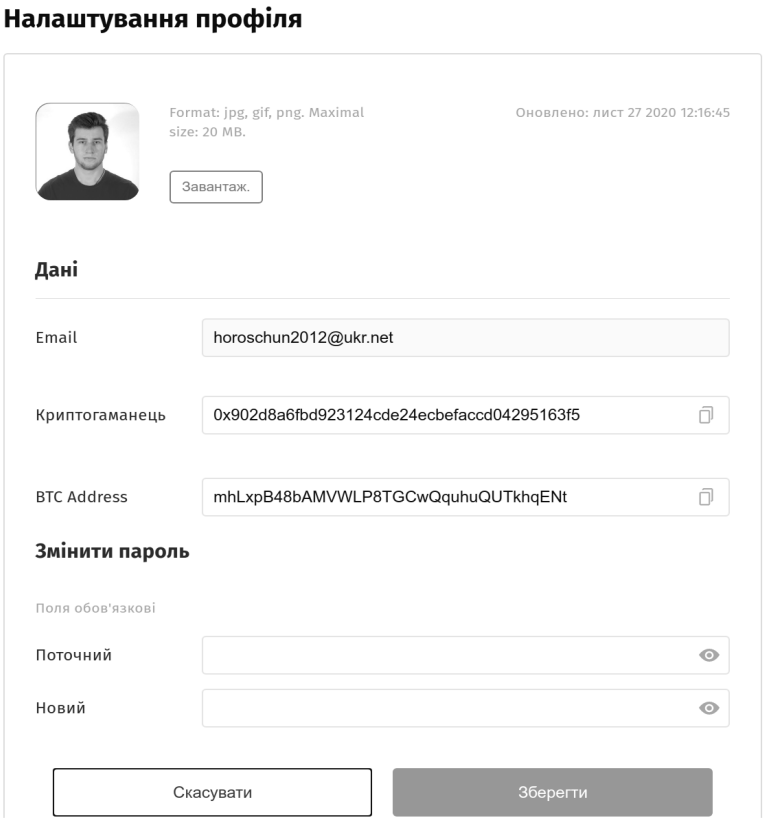


Рисунок Г.5 – Налаштування профілю користувача

На цій сторінці користувач може керувати своїм профілем та встановити адресу криптогаманця. Після встановлення адреси, отриманої на попередньому етапі, та натискання на кнопку збереження, у користувача з’являється можливість створювати запити на проведення обміну. Для цього слід перейти на сторінку обмінної платформи та створити новий запит. Пічля цього кнопка обміну стає доступною і при натисканні на неї з’являється інформаційне вікно обміну (рисунок Г.6).

В цьому вікні відображається загальна інформація про обмін, що здійснюється, а саме, адреса, на яку варто відправляти зовнішню криптовалюту; таймер, який відображає час, що залишився до припинення відслідковування заявки; сума, яку необхідно переказати.

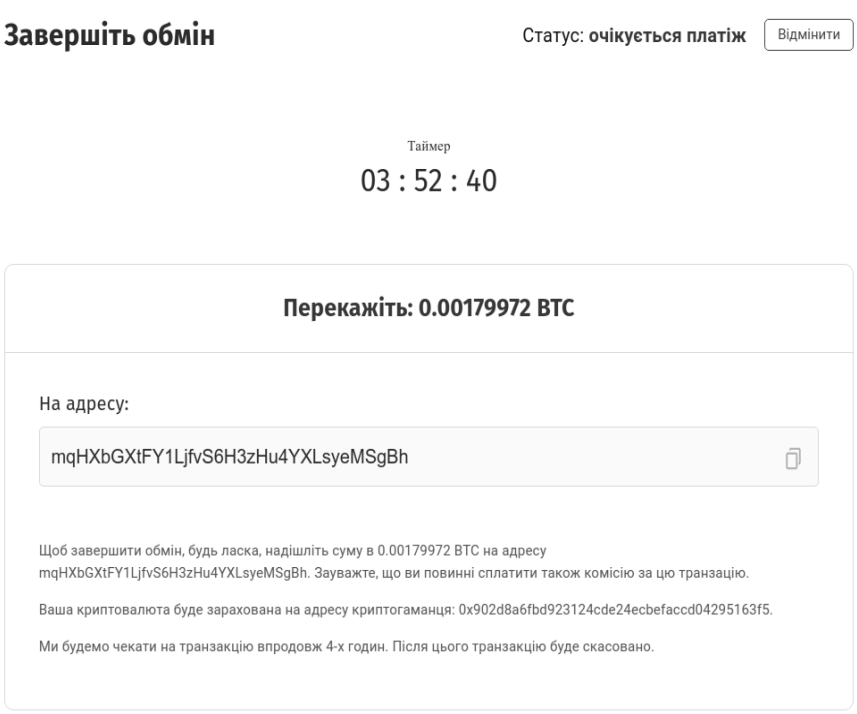


Рисунок Г.6 – Інформаційне вікно обміну

На наступному етапі потрібно перейти у гаманець криптовалюти і зробити переказ на вказану адресу; після цього інформаційне вікно обміну буде закрите, натомість відкриється вікно історії транзакцій з успішним обміном (рисунок Г.7).

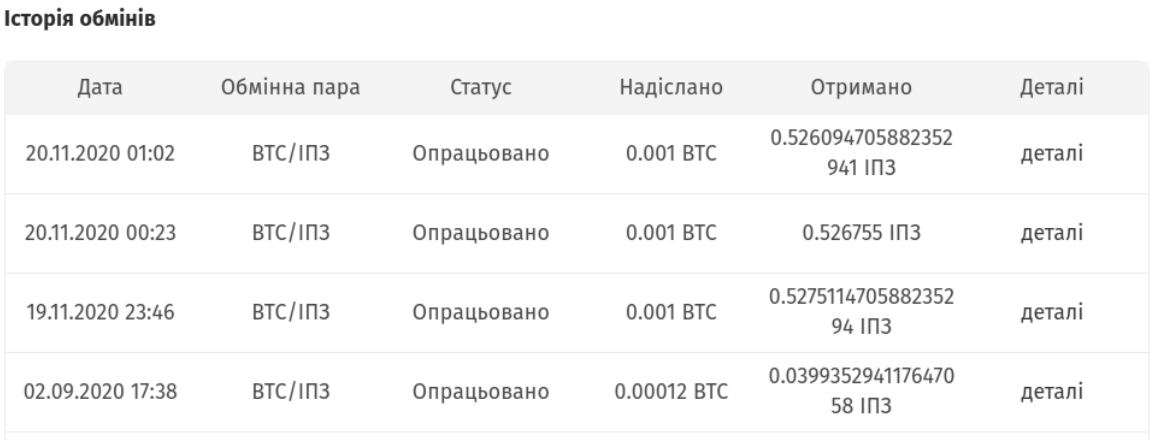


Рисунок Г.7 – Історія обмінів

Після цього варто перейти у свій криптогаманець; при цьому з’явиться інформаційна панель, яка буде показувати інформацію про вхідну транзакцію та баланс (рисунок Г.8).

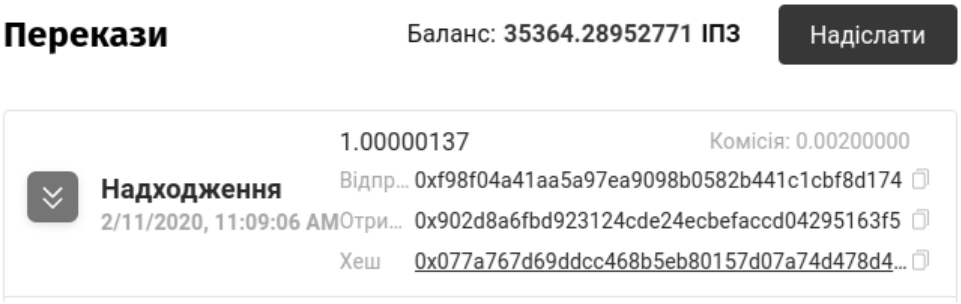


Рисунок Г.8 – Історія транзакцій всередині криптогаманця

Маючи кошти на рахунку, користувач може надсилати їх на рахунки інших користувачів. Після натискання на кнопку надсилання з’явиться вікно надсилання переказу, зображене на рисунку Г.9.

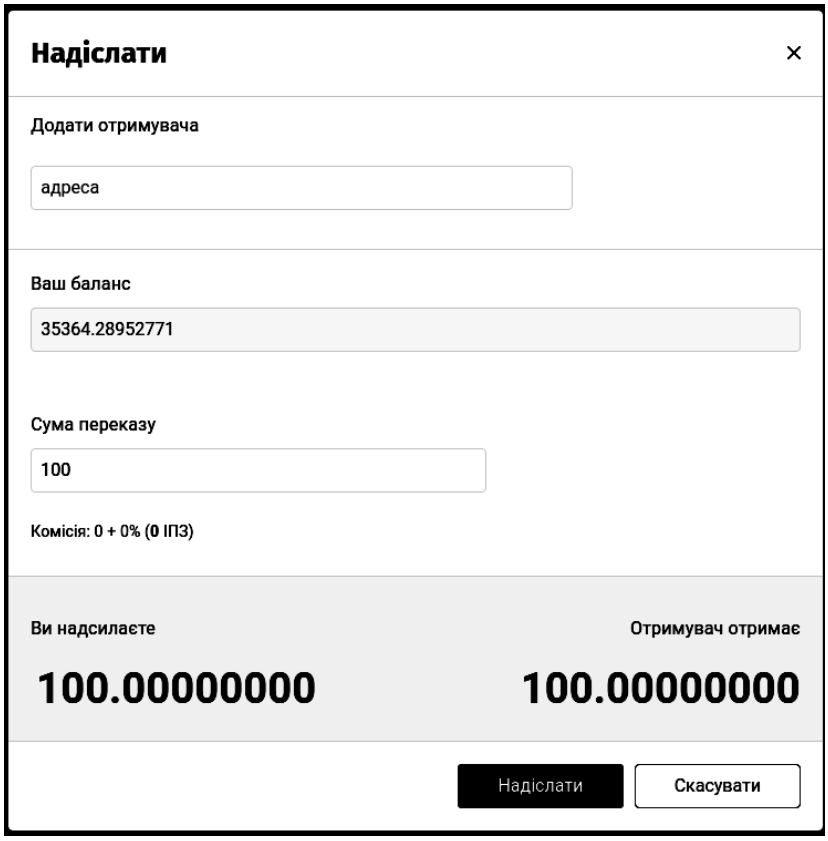


Рисунок Г.9 – Вікно надсилання переказу

У цьому вікні необхідно вказати адресу отримувача та суму переказу. Далі можна ознайомитись з сумою комісії, яка буде знята, та натиснути кнопку

надсилання. Після цього з акаунту користувача вказана сума буде знята та перерахована на рахунок отримувача.

Розглянемо також процедуру процедуру зміни налаштувань системи адміністратором. Для зміни налаштувань адміністратор має пройти авторизацію в адміністративній платформі за процедрую, аналогічною тій, яка відбувалась при вході користувача в іншу частину платформи. Після введення авторизаційних даних адміністратор отримує доступ до вікна налаштувань системи (рисунок Г.10).

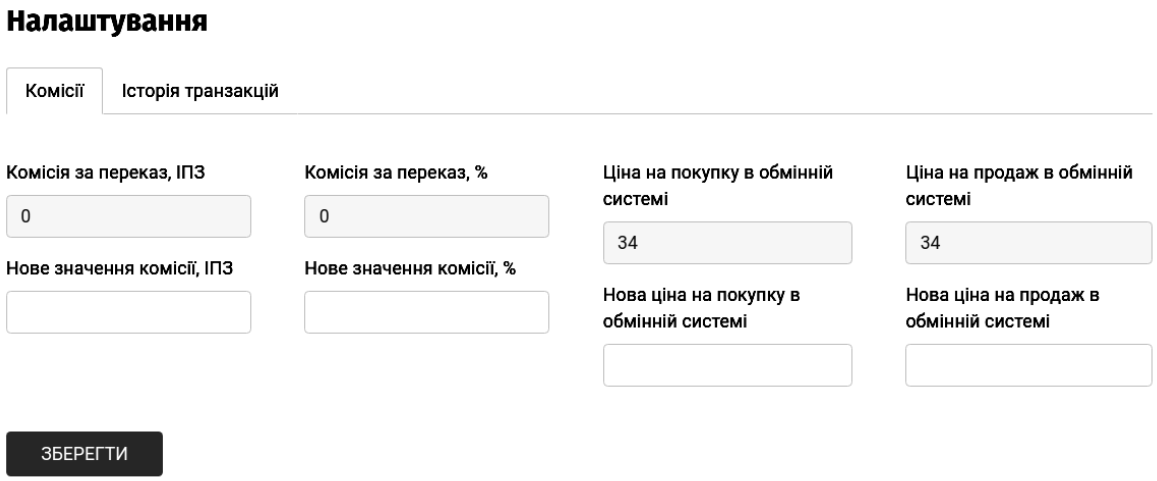


Рисунок Г.10 – Вікно налаштування системи для адміністратора

Для того, щоб змінити налаштування будь-якого значення системи, адміністратору варто вказати нове значення у відповідній комірці та натиснути кнопку збереження. Після обробки запиту зміни будуть впроваджені у системі.

ДОДАТОК Д

(обов’язковий)

КОПІЇ НАУКОВИХ ПУБЛІКАЦІЙ

ISSN 2307-5732
DOI 10.31891/2307-5732

НАУКОВИЙ ЖУРНАЛ **4.2020**

ВІСНИК

Хмельницького національного університету

Том 1

Технічні науки

Technical sciences

SCIENTIFIC JOURNAL
HERALD OF KHMELNYTSKYI NATIONAL UNIVERSITY

2020, Issue 4, Volume 287, Part 1

Хмельницький

ВІСНИК
ХМЕЛЬНИЦЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
серія: Технічні науки

Затверджений як фахове видання (перереєстрація)
Категорія «Б», РІШЕННЯ АТЕСТАЦІЙНОЇ КОЛЕГІЇ № 1643 ВІД 28.12.2019

Засновано в липні 1997 р. ***Виходить 6 разів на рік***

Хмельницький, 2020, № 4, Том 1 (287)

Засновник і видавець: Хмельницький національний університет
(до 2005 р. – Технологічний університет Поділля, м. Хмельницький)

Включено до науково-метричних баз:

Google Scholar	http://scholar.google.com.ua/citations?hl=uk&user=aIUP9OYAAAAJ
Index Copernicus	http://jml2012.indexcopernicus.com/passport.php?id=4538&id_lang=3
Polish Scholarly Bibliography	https://pbn.nauka.gov.pl/journals/46221

Технічний редактор Горященко К. Л., к.т.н.
Редактор-коректор Броженко В. О.

Рекомендовано до друку рішенням вченої ради Хмельницького національного університету, протокол № 3 від 29.10.2020 р.

Адреса редакції: редакція журналу "Вісник Хмельницького національного університету"
Хмельницький національний університет
вул. Інститутська, 11, м. Хмельницький, Україна, 29016
☎ (038-2) 67-51-08 **web:** http://journals.khnu.km.ua/vestnik
e-mail: visnyk.khnu@gmail.com http://lib.khnu.km.ua/visnyk_tup.htm

Зареєстровано Міністерством України у справах преси та інформації.
Свідоцтво про державну реєстрацію друкованого засобу масової інформації
Серія КВ № 9722 від 29 березня 2005 року

ЗМІСТ

КОМП'ЮТЕРНІ НАУКИ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІ, СИСТЕМНИЙ АНАЛІЗ ТА КІБЕРБЕЗПЕКА

МАШИНОБУДУВАННЯ, МЕХАНІКА ТА МАТЕРІАЛОЗНАВСТВО

ЕЛЕКТРОМЕХАНІКА, ЕЛЕКТРОТЕХНІКА ТА ЕНЕРГЕТИКА

АВТОМАТИЗАЦІЯ, ТЕЛЕКОМУНІКАЦІЇ ТА РАДІОТЕХНІКА

DOI 10.31891/2307-5732-2020-287-4-89-93
УДК 004.75:336.744

Г.І. РАДЕЛЬЧУК, М.Л. ХОРОШУН
Хмельницький національний університет

КОНЦЕПЦІЇ ПРОЕКТУВАННЯ ДЕЦЕНТРАЛІЗОВАНОЇ ПЛАТІЖНОЇ СИСТЕМИ З ВЛАСНОЮ ЦИФРОВОЮ ВАЛЮТОЮ НА БАЗІ БЛОКЧЕЙН-ПЛАТФОРМИ ETHEREUM

У роботі представлено концепції проектування децентралізованої платіжної системи з власною криптовалютою на базі блокчейн-платформи Ethereum. У дослідженні удосконалено метод створення цифрових платіжних засобів шляхом проектування комплексного рішення, яке складається з клієнтської частини та системи розумних контрактів. Обгрунтовано доцільність проектування власної обмінної платформи поряд з інтеграцією зовнішніх криптовалютних бірж для можливості купівлі розробленої криптовалюти за інші популярні цифрові валюти. Удосконалено методи опрацювання транзакцій, що дозволило оптимізувати пропускну здатність системи у порівнянні з існуючими рішеннями. Розглянуто алгоритми функціонування блокчейн-систем та обрано оптимальний алгоритм консенсусу. Результатом дослідження є покращені методи проектування децентралізованих платіжних систем.

Ключові слова: блокчейн, децентралізована платіжна система, цифрова валюта.

GALINA IVANIVNA RADELCHUK, MYKHAILO LEONTIIOVICH KHOROSHUN
Khmelnytsky National University

CONCEPTS OF DESIGNING A DECENTRALIZED PAYMENT SYSTEM WITH ITS OWN DIGITAL CURRENCY BASED ON THE BLOCKCHAIN PLATFORM ETHEREUM

The study is devoted to the research of design concepts for the development of decentralized payment systems with its own cryptocurrency based on the blockchain platform Ethereum. The study examines the problems associated with traditional models of decentralized payment systems, analyzes the shortcomings in the industry, and proposes approaches to their solution. During the research, the method of creating digital assets was improved by designing a comprehensive solution consisting of a smart contracts part and a web-client part. Also, the expediency of designing its own exchange platform along with the integration of external cryptocurrency exchanges for the possibility of buying developed cryptocurrency for other popular digital currencies was substantiated and visual schemes of the traditional payment system structure with the method of exchange platform integration were presented. The paper substantiates the use of dynamic block size as a method of optimizing system efficiency, which can increase or decrease the maximum block size at the algorithm level, depending on the number of pending transactions, thus providing the required level of bandwidth. Also, the need for one-to-many transfers is justified, which involves the development of new methods of smart contracts for the cryptocurrency transfer from one user to a large number of addresses within a single transaction in order to reduce the load on the network. The problem of high energy consumption required for the functioning of the consensus system in the traditional model was solved by developing a model based on protocols that do not use the computational power of participants as a parameter to maintain consensus. Thus, the results of this study present improved methods for designing decentralized payment systems. The obtained results and generated recommendations can be used in the design of decentralized payment systems.

Keywords: blockchain, decentralized payment system, digital currency.

Вступ. Постановка проблеми

На сучасному етапі розвитку людства гроші втратили товарну сутність і сприймаються лише як розрахункова одиниця. Але еволюція грошей продовжується – паперові гроші перетворюються на цифрові, і все більше виокремлюються децентралізовані платіжні системи як альтернатива центральним банкам. Децентралізовані платіжні системи дозволяють виключити банківську систему з процесу емісії грошей і проведення транзакцій та довірити це комп'ютерним алгоритмам. Такі системи не мають обмежень у формуванні обмінних курсів та здійсненні операцій та дозволяють виконувати вільне переміщення коштів. Транзакції у таких системах не піддаються цензурі та є незворотніми. Поява децентралізованих платіжних систем продемонструвала, що криптовалюти можуть бути ефективним інструментом інвестування. Випуск віртуальної цифрової валюти, яку далі можна використовувати як платіжний засіб усередині сервісу або компанії, виявився найпростішим і найпривабливішим способом як для залучення інвестицій, так і для інвестування. Таким чином, з'явився попит на створення децентралізованих платіжних систем з власними цифровими валютами. Зазвичай, криптовалюта в таких системах виступає в ролі внутрішньої валюти додатку. Однак, методи реалізації подібних систем різняться за показниками ефективності, безпеки коштів та цінністю валюти на глобальному ринку.

Аналіз останніх досліджень та публікацій

Криптовалюти – одна з найновітніших технологій у сфері фінансів. Сьогодні наукова спільнота активно проводить дослідження у сфері криптовалютних платіжних систем. Серед робіт на тему криптовалют у першу чергу варто відзначити працю [1], у якій автори детально дослідили більше сотні популярних криптовалют та їхні ключові характеристики, а також ситуацію на ринку криптовалют у цілому та на криптовалютних біржах зокрема. У роботі [2] дослідники розглянули ефективність алгоритму консенсусу Proof-of-Work (PoF), який застосовується у більше ніж 90% сучасних децентралізованих систем. У роботі [3] здійснено грунтовний аналіз популярних криптовалют та визначені їхні переваги та недоліки у порівнянні з фіатними валютами.

Виділення невирішених частин загальної проблеми

У передових економіках світу цифрові платіжні системи розвиваються активними темпами, і безумовною перевагою криптовалют є можливість здійснення прямих платежів між користувачами, відсутність національних кордонів для здійснення переказів та зниження операційних витрат у порівнянні з традиційними банківськими системами [4]. Але, окрім внутрішніх витрат за перекази коштів між користувачами мережі для здійснення оплати в криптовалюті, користувачу, який не має її у своєму розпорядженні, необхідно спершу обміняти наявні у нього гроші на криптовалюту, що потребує використання онлайн-бірж або обмінників. Така конвертація валют містить додаткові комісійні витрати. Наприклад, найпопулярніша в Україні онлайн-біржа KUNA встановлює 1,5% комісійних від суми платежу при купівлі криптовалюти за фіатні гроші та 0,25% на обмін криптовалют [5]. При здійсненні платежу у криптовалюті з рахунку на біржі встановлюються фіксовані комісійні. Таким чином, при здійсненні платежів із використанням криптовалют транзакційні витрати всередині децентралізованої платіжної системи можуть бути досить низькими та, враховуючи можливості транскордонного переміщення коштів, більш привабливими у порівнянні із банківськими платежами. Але якщо користувачу необхідно здійснювати обмін або купувати криптовалюту для здійснення такого платежу, то додаткові комісійні витрати можуть перевищувати аналогічні витрати у банківській системі.

Також однією з головних проблем для децентралізованих платіжних систем залишається масштабованість. Наприклад, у рамках протоколу Bitcoin блок транзакцій обмежений розміром в 1 мегабайт і швидкість їх обробки становить приблизно сім операцій в секунду (у той же час Visa обробляє в середньому 2000 операцій за секунду). Розмір блоку впливає на кількість транзакцій, які можна додати у блок. Протокол Bitcoin передбачає, що блок формується в середньому 10 хвилин, і при збільшення активності у мережі збільшуються як комісійні, що пропонуються відправниками, так і час підтвердження окремої транзакції вузлами мережі.

Ще одним важливим моментом при здійсненні платежів є спосіб підтвердження транзакцій. При використанні централізованих банківських платіжних систем банк виступає посередником і гарантом переміщення коштів між рахунками клієнтів. У децентралізованих платіжних системах визначення, чи є транзакція вірною, відбувається на основі консенсусу учасників такої системи (тобто її підтвердження здійснюється «більшістю голосів»). Понад 90% існуючих систем використовують алгоритм консенсусу Proof-of-Work [2]. Суть цього алгоритму зводиться до двох основних пунктів:

- необхідності виконання певного, досить складного і тривалого завдання;
- можливості швидко і легко перевірити результат.

Необхідність постійного розрахунку рішення робить вирішення задачі дуже ресурсномістким, у зв`язку з чим десятки тисяч комп`ютерів витрачають власні обчислювальні ресурси на виконання протоколу консенсусу і при цьому лише один з них наприкінці отримує можливість створити блок. У результаті це призводить до великих енергозатрат, що є проблемою.

Таким чином, проаналізувавши наукові дослідження та публікації, можна зробити висновок, що основними проблемами, пов'язаними з існуючими децентралізованими платіжними системами є наступні: низька швидкість проходження транзакцій; необхідність залучення третьої сторони (криптовалютних бірж) для купівлі/обміну власної криптовалюти; функціонування системи на алгоритмі консенсусу Proof-of-Work, що вимагає значних енергозатрат. Тому основною задачею даного дослідження є покращення методів створення децентралізованих систем задля вирішення існуючих проблем.

Формулювання цілей

Для проведення дослідження сформульовано наступні цілі: провести теоретичний аналіз процедур функціонування децентралізованих платіжних систем; охарактеризувати базову модель організації процесів функціонування децентралізованих платіжних систем; описати існуючі механізми реалізації децентралізованих платіжних систем; виділити наявні проблеми в галузі та описати шляхи їх вирішення; покращити існуючі алгоритми функціонування децентралізованих систем.

Виклад основного матеріалу

Децентралізована платіжна система – цифрова пірингова платіжна система, яка використовує криптовалюту як розрахункову одиницю для обліку операцій. Така комп'ютерна мережа заснована на рівноправ'ї учасників (тобто відсутні виділені сервери), а кожний вузол є як клієнтом, так і виконує функції сервера. На відміну від архітектури «клієнт-сервер» така організація дозволяє зберігати працездатність мережі при будь-якій кількості та будь-якому поєднанні доступних вузлів. Функціонування та захист системи забезпечуються використанням криптографічних методів. При цьому вся інформація про транзакції між адресами системи доступна у відкритому вигляді. Серед популярних децентралізованих платіжних систем можна виділити мережі Bitcoin, Litecoin, Ethereum, Stellar, серед яких найпоширенішою є Ethereum.

Традиційні децентралізовані платіжні системи складаються з двох архітектурних компонентів. Основна логіка роботи системи працює на блокчейн-платформі (наприклад, на розумних контрактах). Для забезпечення зручних інтерфейсів розробляється веб-частина, до складу якої найчастіше входять наступні інструменти: оглядач блоків і транзакцій для показу статистичної та службової інформації про платіжну систему; відділ адміністрування, де здійснюється керування платформою; криптогаманець, за допомогою якого кінцевий користувач здійснює операції надсилання криптовалютних коштів чи отримує інформацію про їх отримання.

При такій формі організації системи існує очевидна проблема − після закінчення емітування монет у мережі користувачі можуть отримати криптовалюту, лише купивши її в інших учасників системи або на криптовалютних біржах. Це призводить до появи небажаних комісій та додаткових ризиків для користувачів, які вимушені користуватись сторонніми додатками для придбання криптовалюти. Рішенням цієї проблеми може стати розробка власної обмінної платформи у комплексі однієї платіжної системи. Таким чином, користувач матиме вибір − здійснювати переказ через зовнішні біржі на ринку чи скористатись офіційною обмінною платформою. В той час, як на офіційній обмінній платформі ціна на криптовалюту буде вищою за ринкову, користувачі, які нею користуватимуться, будуть впевнені у безпеці проведення своїх операцій, оскільки їм не потрібно покладатись на сторонні платформи. На рис. 1 наочно продемонстровано схему платформи і те, як компонент обмінної системи інтегрується в неї.

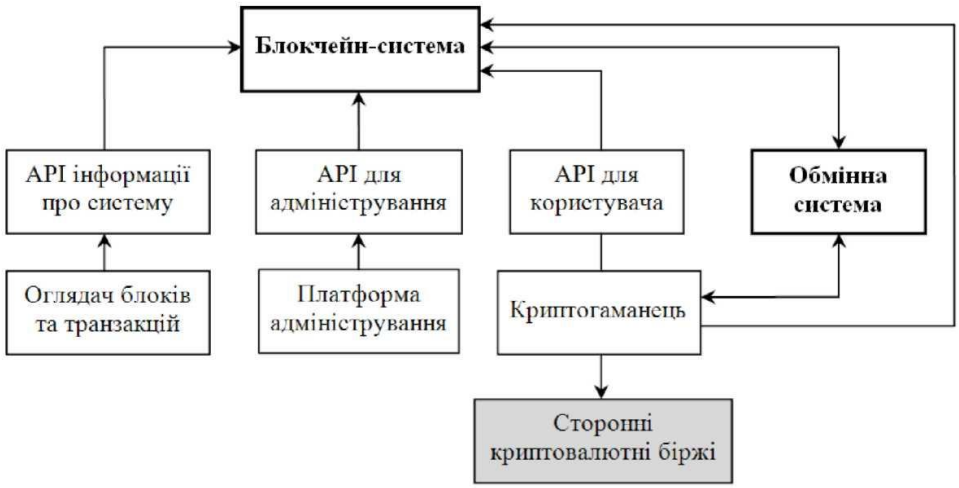


Рис. 1. Взаємодія компонентів системи

Розглянемо детальніше компоненти на представленій схемі. Очевидно, що основним її компонентом є блокчейн-система, яку використовують інші частини платформи для виконання операцій. Блокчейн-система реалізовує головний функціонал системи − збереження даних про криптовалюту та користувацькі рахунки, емітування монет, опрацювання переказів між користувачами. Інші компоненти системи лише звертаються до методів розумних контрактів для отримання даних чи для виконання певних операцій.

В той час, як операції для отримання даних з блокчейну відбуваються за допомогою звичайних методів читання бази даних смарт-контрактів (які миттєво повертають результат), виконання операцій на модифікацію існуючих даних (таких, наприклад, як перекази між рахунками) відбуваються за визначеним алгоритмом (рис. 2). Всі транзакції у системі зберігаються у спеціальних структурах даних − блоках. Під час роботи системи через деякий визначений часовий інтервал у блокчейн додається новий блок. Цей блок може бути порожнім або містити інформацію про деяку кількість транзакцій, обмежену розміром блоку. Кожний наступний блок зберігає посилання на попередній, формуючи таким чином єдиний нерозривний ланцюг. При додаванні нового блоку до ланцюга, транзакції, що містяться в ньому, почергово виконуються на розумних контрактах.

Очевидно, що з таким алгоритмом опрацювання транзакцій неможливо здійснювати горизонтальне масштабування системи. Незважаючи на те, що блокчейн є розподіленою системою, у підсумкову базу даних записується лише один блок, від одного вузла, тому що опрацювання блоків може здійснювати лише один канал обслуговування. Оскільки збільшення кількості каналів обробки є неможливим, то для підвищення пропускної здатності системи у роботі запропоновані наступні методи оптимізації.

1 Використання динамічного розміру блоку. Кожний блок у системі має фактичний та максимальний розмір. Фактичний розмір блоку відповідає об'єму транзакцій, які в ньому знаходяться. Максимальний розмір блоку відповідає максимальному значенню сукупного розміру транзакцій, які можуть розміститися в ньому, та визначається системою під час її старту. Передбачивши можливість збільшення максимального розміру блоку вже після запуску мережі, можна масштабувати пропускну здатність системи: при збільшенні навантаження на систему значення максимального розміру блоку буде зростати, а при зменшенні навантаження − зменшуватись. Таким чином, з'явиться змога опрацьовувати більшу кількість транзакцій за той самий часовий період. Реалізація цього функціоналу полягатиме у підтриманні консенсусної зміни конфігурацій. Створювачу блоків має бути надана можливість змінювати максимальний розмір для свого блоку, але не більше ніж на 1% від розміру попереднього блоку. Таким чином, значення максимального розміру блоку в системі відповідатиме навантаженню на систему.

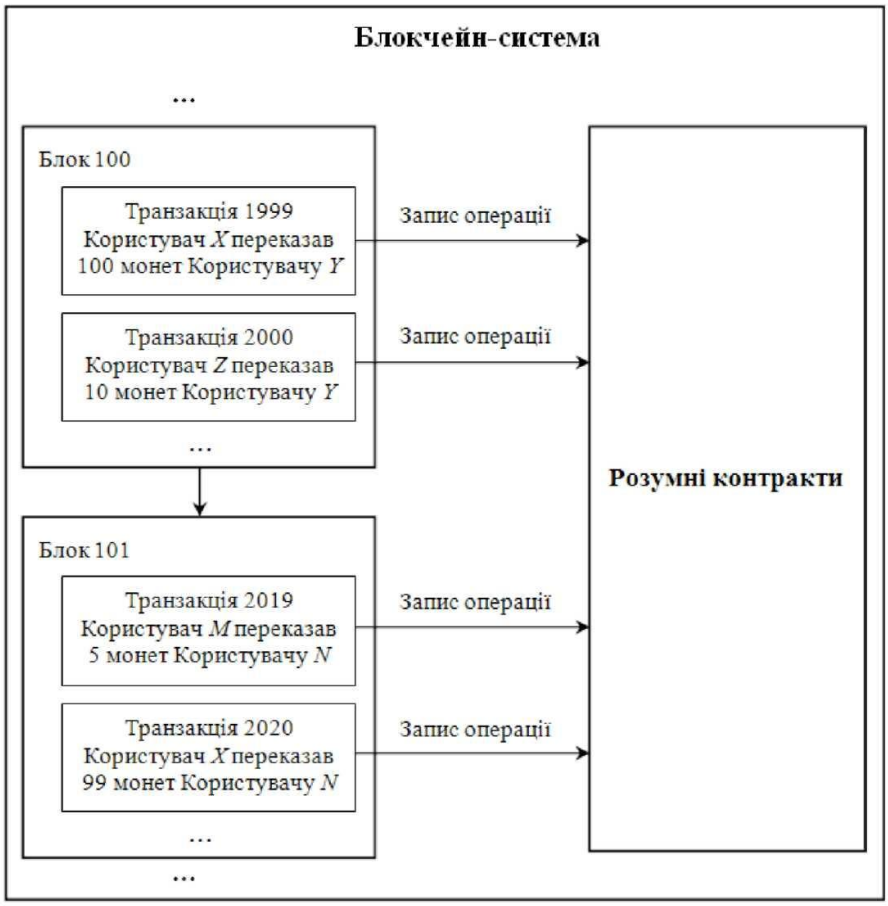


Рис. 2. Опрацювання транзакцій всередині блокчейн-системи

2 Перекази «один до багатьох». Також для того, щоб зменшити навантаження на мережу, доцільно розробити функціонал виклику трансферу криптовалюти, при якому в якості вхідних аргументів методу смарт- контракту можна було б передати список адрес отримувачів та необхідні суми переказу кожному з отримувачів. Таким чином, при здійсненні переказу на значну кількість адрес, користувач зможе це зробити всередині лише однієї транзакції, що зменшить навантаження на мережу.

У процесі застосування технології блокчейн виникає також питання підбору ефективного протоколу консенсусу. Консенсус блокчейна полягає в тому, що всі вузли підтримують однаковий розподілений реєстр. У традиційній архітектурі програмного забезпечення це не є проблемою через існування центрального сервера, з яким узгоджуються інші вузли. Однак, у розподіленій мережі (такій як блокчейн) кожен вузол є і хостом, і сервером, і, щоб досягти консенсусу, йому потрібно обмінюватися інформацією з іншими вузлами. Інколи деякі вузли можуть працювати у режимі офлайн. Окрім того, можуть з'явитись деякі шкідливі вузли, які будуть негативно впливати на процес консенсусу і, навіть, можуть зашкодити йому. Тому потрібен протокол, який не допустить виникнення подібних ситуацій та мінімізує негативний вплив шкідливих вузлів таким чином, щоб вони не впливали на кінцевий результат консенсусу.

Аналіз предметної області показав, що близько 90% існуючих децентралізованих систем використовують алгоритм консенсусу Proof-of-Work. Однак, значним недоліком PoW-алгоритму є те, що його функціонування потребує постійних значних затрат електроенергії на процес створення блоків. Цей недолік яскраво видно на прикладі Біткоіна, який реалізовує вказаний алгоритм. За один рік на функціонування мережі витрачається більше електроенергії, ніж використовує Швейцарія за цей же період. Проблему значних енергозатрат можна вирішити шляхом розробки моделі на основі протоколів, що не використовують обчислювальну здатність учасників як параметр підтримання консенсусу. У роботі [6] автори здійснили порівняльний аналіз найпопулярніших алгоритмів консенсусу, розглянувши методи обчислення значень основних характеристик алгоритмів: відмовостійкість, ресурсоємність, масштабованість та придатність для публічних мереж. На основі результатів цього аналізу сформовано порівняльну таблицю 1.

Таблиця 1

Характеристика різних алгоритмів консенсусу

Характеристика	PoW	PoS	PBFT	Ripple
Тип алгоритму	Ймовірнісно–кінцевий	Ймовірнісно–кінцевий	Абсолютної остаточності	Абсолютної остаточності
Відмовостійкість	50%	50%	33%	20%
Ресурсоємність	Висока	Середня	Низька	Низька
Масштабованість	Добра	Добра	Погана	Погана
Придатність для публічних мереж	Придатний	Придатний	Не придатний	Придатний

Як бачимо з табл. 1, серед алгоритмів, придатних для публічної мережі, алгоритми PoW і PoS (Proof of Stake) відрізняються лише за показником ресурсоємності. При цьому PoS набагато менш ресурсоємний, тому він є кращим вибором для публічного блокчейну. Вибір цього алгоритму консенсу для системи дозволить зменшити її ресурсоємність та забезпечить необхідними інструментами персоналізації.

Висновки

Таким чином, у роботі досліджено процедури залучення криптовалютних інвестицій, охарактеризовано структуру предметної області та виділено основні невирішені проблеми. Розглянуто та описано моделі, за якими працюють традиційні децентралізовані платіжні системи. Запропоновано методи та рішення, які дозволять оптимізувати їхню роботу: підвищити швидкість проходження транзакцій; забезпечити відсутність необхідності залучення «третьої» сторони (сторонніх криптовалютних бірж) для купівлі/продажу власної криптовалюти; оптимізувати використання енергозатрат для підтримки функціонування системи шляхом вибору та застосування ефективного алгоритму консенсусу.

Література

1. Hileman G. Global cryptocurrency benchmarking study / G. Hileman, M. Rauchs. – Cambridge, 2017. – 114 p.

2. Gervais A. On the Security and Performance of Proof of Work Blockchains / A. Gervais, G. Karame, K. Wüst etc. // CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. – 2016. – № 10. – P. 3–16.

3. Dorofeyev M. Trends and Prospects for the Development of Blockchain and Cryptocurrencies in the Digital Economy / M. Dorofeyev, M. Ksov, V. Ponkratov // European Research Studies Journal. – 2018. – № 21. – P. 429–445.

4. Олешко А. А. Інноваційні тенденції розвитку безготівкової економіки / А. А. Олешко // Інвестиції: практика та досвід. – 2018. – № 10. – С. 22–25.

5. Бойко В. П. Переваги та недоліки використання децентралізованих платіжних систем як інноваційного способу транскордонних розрахунків / В. П. Бойко // Інвестиції: практика та досвід. – 2019. – № 8. – С. 75–82.

6. Золотарьова І. О. Інформаційні технології оптимізації роботи приватного блокчейн за допомогою вибору алгоритму консенсусу / І. О. Золотарьова, Г. О. Плеханова // Системи обробки інформації. – 2020. – № 1. – С. 107–114.

References

1. Hileman G. Global cryptocurrency benchmarking study / G. Hileman, M. Rauchs. – Cambridge, 2017. – 114 p.

2. Gervais A. On the Security and Performance of Proof of Work Blockchains / A. Gervais, G. Karame, K. Wüst etc. // CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. 2016. № 10. P. 3 16.

3. Dorofeyev M. Trends and Prospects for the Development of Blockchain and Cryptocurrencies in the Digital Economy / M. Dorofeyev, M. Ksov, V. Ponkratov etc. // European Research Studies Journal. – 2018. – № 21. – P. 429–445.

4. Oleshko A. A. Innovatsiyni tendentsiyi rozvytku bezhotivkovoyi ekonomiky / A. A. Oleshko // Investytsiyi: praktyka ta dosvid. – 2018. № 10. S. 22 25.

5. Boyko V. P. Perevahy ta nedoliky vykorystannya detsentralizovanykh platizhnykh system yak innovatsiynoho sposobu transkordonnykh rozrakhunkiv / V. P. Boyko // Investytsiyi: praktyka ta dosvid. 2019. № 8. S. 75 82.

6. Zolotar'ova I. O. Informatsiyni tekhnolohiyi optymizatsiyi roboty pryvatnoho blokcheyn za dopomohoyu vyboru alhorytmu konsensusu / I. O. Zolotar'ova, H. O. Plekhanova // Systemy obrobky informatsiyi. 2020. № 1. S. 107 114.

Рецензія/Peer review : 10.10.2020 р. Надрукована/Printed :02.11.2020 р.

MATERIÁLY
XVI MEZINÁRODNÍ VĚDECKO - PRAKTICKÁ KONFERENCE

VĚDECKÝ PRŮMYSL EVROPSKÉHO KONTINENTU - 2020

22 - 30 listopadu 2020 r.

Volume 6

Praha
Publishing House «Education and Science»
2020

Vydáno Publishing House «Education and Science»,
Frýdlanská 15/1314, Praha 8
Spolu s DSP SHID, Berdianskaja 61 B, Dnepropetrovsk

Materiály XVI Mezinárodní vědecko - praktická konference «Vědecký průmysl evropského kontinentu - 2020», Volume 6 : Praha. Publishing House «Education and Science» -104 s.

Šéfredaktor: Prof. JUDr Zdenák Černák

Náměstek hlavního redaktora: Mgr. Alena Pelicánová

Zodpovědný za vydání: Mgr. Jana Štefko

Manažer: Mgr. Helena Žákovská

Technický pracovník: Bc. Kateřina Zahradníková

Materiály XVI Mezinárodní vědecko - praktická konference ,
Vědecký průmysl evropského kontinentu - 2020

Pro studentů, aspirantů a vždeckách pracovníků

Cena 50 kč

ISSN 1561-6940

ZEMĚDĚLSTVÍ

Storage Technology a zpracování zemědělských produktů

MODERNÍ INFORMAČNÍ TECHNOLOGIE

Software

Informační bezpečnost

TECHNICKÉ VĚDY

Mechanika

Větev inženýrství

MATEMATIKA

Vědecký průmysl evropského kontinentu - 2020 ★ Volume 6

MODERNÍ INFORMAČNÍ TECHNOLOGIE

Software

К. т. н. Радельчук Г. І., Хорошун М. Л.

Хмельницький національний університет, Україна

ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ ЗАЛУЧЕННЯ КРИПТОВАЛЮТНИХ ІНВЕСТИЦІЙ НА БАЗІ БЛОКЧЕЙН-ПЛАТФОРМИ ETHEREUM: КОНЦЕПТУАЛЬНІ ЗАСАДИ

У січні 2009 року відбулась революція у фінансовій сфері. Анонімний дослідник під псевдонімом «Сатоші Накамото» створив першу децентралізовану однорангову систему електронних платіжних засобів «Bitcoin».

У 2015 році відбулась ще одна революція у криптовалютній сфері – світ побачила платформа Ethereum, на основі якої стало можливим створювати власні криптовалюти. Це призвело до появи нових форм залучення інвестицій.

Через складні процедури та високі вимоги верифікації люди середньої ланки не могли стати інвесторами. Після запуску Ethereum ця ситуація змінилась – завдяки появі первинного розміщення монет ринок інвестицій став доступним для широкого кола осіб. Мінімальні регулювання від представників державних комісій створили сприятливі умови для успішного інвестування.

Таким чином, з'явився попит на реалізацію програмних систем (ПС), які б дозволяли проводити такі форми залучення коштів.

Первинне розміщення монет (Initial Coin Offering – ICO) – це один із найсучасніших способів збору коштів на фінансування розвитку бізнесу або для запуску нового проекту. По справжньому надійно, як інструмент інвестування, розміщення ICO здійснюється на певних ресурсах – блокчейн-платформах. Це одночасно гарантує вартість монет та можливість їх взаємнообміну. Для кожної мережі існує свій набір правил використання та взаємозамінності цифрового

активу. Будь-яке ICO функціонує на технології блокчейн, а сама прозорість логіки його функціонування досягається шляхом написанням спеціальних програм – «розумних контрактів», які здійснюють контроль над інвестиціями.

Розумний контракт (Smart Contract) – це комп'ютерний протокол, призначений для цифрового укладення, зміни, виконання та розірвання угод. Розумні контракти дозволяють здійснювати ці операції надійно і без сторонніх осіб. Взаємодія з контрактом є відстежуваною та незворотньою. Метою розумних контрактів є забезпечення вищого рівня безпеки, порівняно з традиційними договірними правами, а також зменшення транзакційних витрат.

Сьогодні найпопулярнішою блокчейн-платформою для проведення первинного розміщення монет є загальнодоступна розподілена обчислювальна платформа Ethereum. Вона реалізована як єдина децентралізована віртуальна система, що дозволяє виконувати розумні контракти. Програми, побудовані на цій платформі, називають децентралізованими додатками (Decentralized Applications), оскільки вони базуються на децентралізованій віртуальній машині Ethereum та її розумних контрактах. Це означає, що вся інформація про транзакції синхронізово зберігається в усіх вузлах мережі, які фізично можуть знаходитися далеко один від одного. Таким чином, виключається можливість шахрайства чи несанкціонованої зміни даних сторонніми особами.

Розглянемо детальніше процедуру та регламент проведення ICO.

Попередній анонс. Випускається так званий «білий аркуш» (White Paper) – документ, у якому описуються усі деталі ICO. На цьому етапі визначається цільова аудиторія потенційних інвесторів, оформлюється відповідний документ, у якому описуються всі умови та терміни, які супроводжують процес емісії монет. Обов'язковою умовою є поява офіційного сайту проекту, де будь-хто може ознайомитися з інформацією та дізнатися останні новини компанії.

Попередній продаж монет (Pre-sale ICO або pre-ICO). Часто після виконання юридичних формальностей проводиться закритий попередній продаж монет. У деяких інвесторів з'являється ексклюзивне право придбати цифрові монети за спеціальною ціною зі значною знижкою.

Маркетинговий етап. Коли організатор практично готовий запустити

ICO, починається рекламна кампанія та більш поглиблене ознайомлення публіки з ідеями проекту та його продуктом. Оголошується точна дата продажів та інші важливі умови, які повинні знати всі потенційні інвестори.

Запуск ICO. Запуск ICO – старт продажів. Це обмежений в часі етап, коли безпосередньо реалізовуються монети. На цьому етапі встановлюється сума грошей чи інших платіжних засобів, яку планується зібрати або яка необхідна для купівлі всіх випущених цифрових монет.

Кінцевими користувачами ICO є його інвестори – особи, які вирішили вкласти в нього власні кошти. З точки зору інвестора, основні вимоги, які можна поставити до проведення ICO, це, передусім, вимоги до безпеки вкладень, гарантії повернення інвестицій у разі провалу ICO та забезпечення зручного інтерфейсу взаємодії з розумними контрактами через веб-інтерфейс. Додатковими плюсами також буде розробка монети у стандарті ERC-20, який уже інтегрований у більшість електронних криптогаманців, та забезпечення можливості взаємодії з контрактом напряму, тобто без веб-інтерфейсу.

Загальна схема базової системи з проведення ICO зображена на рисунку 1.

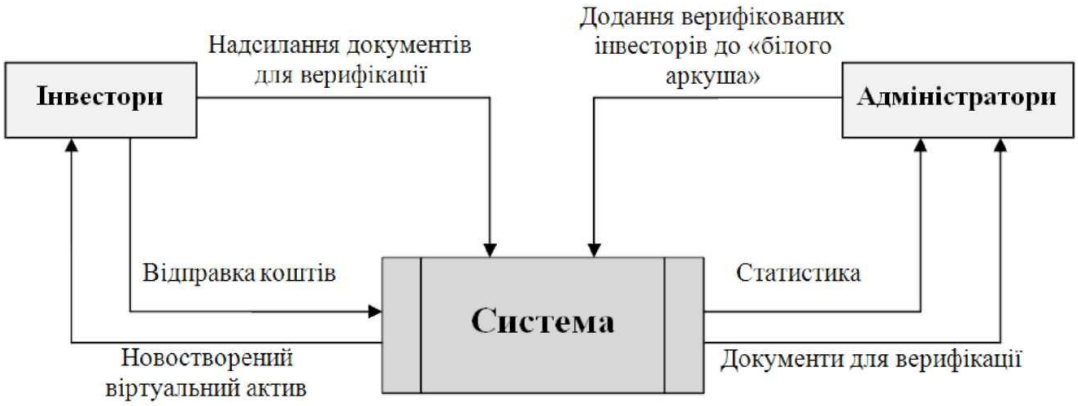


Рисунок 1 – Загальна схема базової системи з проведення ICO

На основі сформованого уявлення про суть ICO та його характеристики можна визначити проблеми, що наразі існують у сфері криптовалютних інвестицій, та накреслити шляхи їх вирішення.

На сьогодні переважна кількість організаторів ICO довіряють розробку

технічних складових ICO так званим блокчейн-агентствам. Оскільки вимоги до різних ICO зазвичай бувають досить специфічними, на ринку досі не присутні загальноприйняті програмні рішення для їх проведення. Найчастіше кожна компанія, що надає послуги зі створення ICO, розробляє всі компоненти «з нуля» або користується власними доробками, які не потрапляють у відкритий доступ. Основними критеріями, якими керується замовник при пошуку блокчейн- агентства для проведення ICO, є вартість його проведення, репутація блокчейн-агентства та його портфоліо готових проектів.

З часу появи ICO ринок IT-компаній, які надають послуги з розробки та проведення ICO, стрімко зростав. Різке збільшення кількості компаній призвело до того, що на ринок потрапили спеціалісти без належного досвіду з розробки розумних контрактів. Це, у свою чергу, спричинило зниження якості програмного коду у загальному по ринку та випадки з виявленням вразливостей у контрактах, якими керувались продажі. Окрім того, існуючі програмні рішення є вузькоспрямованими, налаштованими під конкретні проекти.

Таким чином, у нинішніх умовах ринку існують проблеми невиправдано високих цін на програмні продукти для проведення ICO та небажання компаній розробляти універсальні ПС у зв'язку зі зменшенням потреб ринку у їхніх послугах. При цьому спостерігається тенденція до погіршення якості коду у контрактах ICO, що призводить до зниження показників їх безпеки. Отже, на сьогодні існує потреба у розробці універсальної ПС, яка б акумулювала більшість функцій, необхідних для успішного проведення ICO: гарантію інвестицій, інтеграцію зі стандартом ERC-20, високий рівень безпеки, забезпечення вимог юридичного регулювання тощо. Водночас ця система повинна забезпечити можливість взаємодіяти з ICO через зручний веб-інтерфейс та напряму через розумний контракт.

На основі аналізу предметної області можна визначити та описати вимоги до створюваної ПС.

Основною метою розробки ПС є успішне проведення збору коштів, у результаті якого фінансові активи користувачів за спеціальним курсом будуть

обмінені на новостворену криптовалюту. Після завершення продажів криптовалюта продовжить існувати як самостійний актив, а організатор ICO отримає зібрані кошти та базу даних зацікавлених інвесторів.

У відповідності до встановленої мети розроблювану ПС необхідно забезпечити всіма основними функціями, яких зараз потребує ринок.

На основі аналізу існуючих економічних моделей для розроблюваної ПС обрана модель утилітних монет (Utility Tokens).

В системі існують дві ролі кінцевих користувачів: інвестори та адміністратори. Оскільки з ПС будуть взаємодіяти користувачі різного рівня комп'ютерної грамотності, слід забезпечити максимально інтуїтивний процес роботи зі зручним, зрозумілим інтерфейсом. У зв'язку з цим складну процедуру «знай свого клієнта» (Know Your Customer – KYC) доцільно представити для користувача у вигляді деякої послідовності завдань з підказками, які він повинен виконати. Також процес підтвердження транзакції краще зробити автоматичним, щоб звільнити користувача від будь-яких додаткових дій.

Основне завдання адміністраторів: верифікація користувачів. Це передбачає перевірку правильності введених користувачем даних та їх відповідність до зображень. У зв'язку з можливими недоглядом через «людський фактор» потрібно особливо відповідально поставитись до розробки функціоналу перевірки. Рішенням може бути необхідність ставити «прапорці» (Check-Boxes) біля кожного перевіреного поля, щоб жодне з них не могло бути залишене без уваги через неуважність. Також адміністратори повинні мати доступ до загальної статистики ICO та доступ до історії транзакцій користувачів.

На основі сформованих вимог опишемо «акторів» системи (таблиця 1).

Беручи до уваги додаткові вимоги щодо доступу до внутрішньої функціональності ПС та форм взаємодії з нею, а також з метою формування детальнішого опису вимог, виділимо основні варіанти використання (ВВ) ПС.

Опис варіантів використання наведено у таблиці 2.

Діаграма варіантів використання наведена на рисунку 2.

Таблиця 1 – Опис акторів

Актор	Короткий опис
Інвестор	Розміщує запит на верифікацію з необхідними даними, включаючи графічні. Здійснює фінансовий переказ криптовалюти. Запрошує інших користувачів (завдяки реферальному посиланню). Відстежує власну історію транзакцій. Відстежує бонуси, що надійшли від транзакцій рефералів. Відстежує загальний хід проведення продажів.
Адміністратор	Отримує детальну статистику ходу продажів. Переглядає інформацію про всіх користувачів в системі. Переглядає інформацію про всі транзакції в системі. Верифікує дані користувачів.

Таблиця 2 – Опис варіантів використання ПС

Актори	Найменування ВВ	Опис ВВ
Незареєстрований інвестор	Реєстрація	Створення облікового запису
Зареєстрований інвестор (неверифікований)	Подача запиту на верифікацію	З метою проходження процедури KYC до системи подаються особисті дані інвестора, а також фотографії підтверджуючих документів
Зареєстрований користувач	Авторизація	Авторизація у системі за даними облікового запису
	Перегляд загальної інформації про перебіг продажів	Перегляд інформації про монету, кількість зібраних коштів та час до завершення
Зареєстрований інвестор	Перегляд особистого реферального посилання	Можливість переглянути посилання, що використовується для залучення рефералів
Зареєстрований інвестор (верифікований)	Обмін інвестицій на криптовалюту	Відправка коштів на рахунок продажів з отриманням відповідної кількості криптовалюти на власний рахунок
	Перегляд інформації про особисті транзакції	Можливість перегляду детальних даних про кожну транзакцію
Адміністратор	Перевірка правильності даних користувача	Допуск на запит верифікації користувача (або відмова) в залежності від коректності даних
	Перегляд детальної статистики	Перегляд інформації про всіх зареєстрованих в ПС користувачів та про здійснені ними транзакції

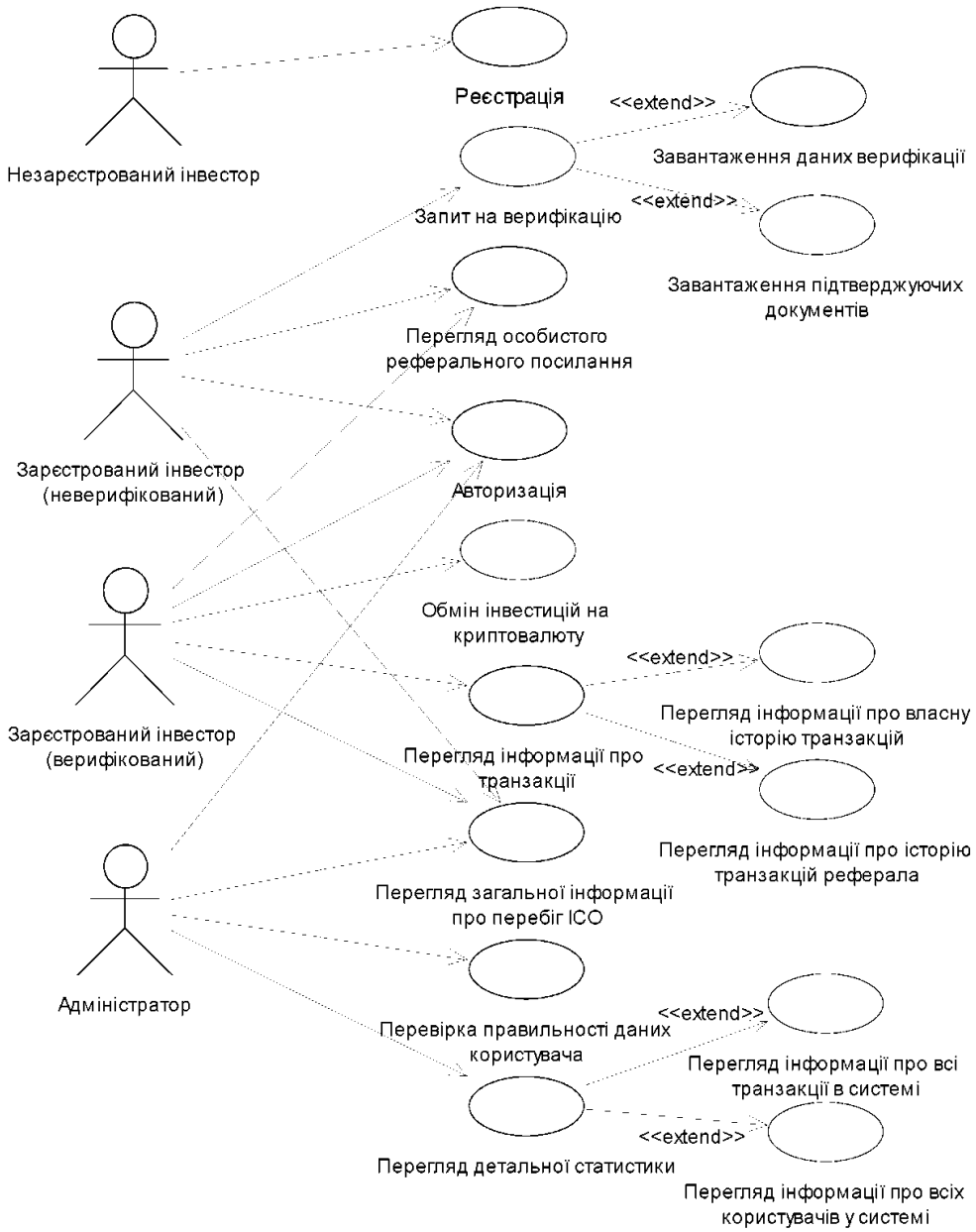


Рисунок 2 – Загальна діаграма варіантів використання системи

Сформуємо також наступні вимоги до самої монети:

– забезпечення емісії (під час ICO для кожної покупки має генеруватись відповідна кількість віртуальної валюти);

– взаємозамінність валюти (всі її одиниці мають бути еквівалентними);

– забезпечення обігу (у валюті мають бути реалізовані можливості передачі її іншому власнику або надання права нею розпоряджатись);

– незалежність валюти (функціонування валюти має бути незалежним від будь-яких інших компонентів системи);

– децентралізованість (після випуску валюти ніхто, включно з її створювачем, не може впливати на її характеристики, закладені при створенні).

Також визначено наступні задачі, які повинна забезпечувати розроблювана децентралізована ПС: створення нової віртуальної валюти; купівля валюти за інший віртуальний актив; обмін валютою між користувачами; реферальна система; зручний веб-інтерфейс взаємодії із системою; реєстрація та авторизація користувачів; процедура «знай свого клієнта»; взаємодія з ПС напряму, без веб-інтерфейсу; надання інформації про історію транзакцій користувача, хід продажів, інформації про валюту (розмір емісії, ціна, назва тощо).

Таким чином, на основі дослідження предметної області криптовалютного інвестування отримано матеріал, у результаті аналізу якого визначено вимоги до ПС та виділено завдання, що підлягають реалізації.

Література

1. Момот І. О. Сутність та особливості функціонування криптовалют / І. О. Момот, Ю. Г. Момот, Д. Є. Козенков. // Економіка і суспільство. – 2018. – № 15. – С. 713–719.

2. Гринчук Д. Р. Перспективи впровадження блокчейн-технологій у бізнесі [Електронний ресурс] / Д. Р. Гринчук, М. О. Чупріна // Збірник наукових праць «Сучасні підходи до управління підприємством». – 2019. – Режим доступу: http://spu.fmm.kpi.ua/article/download/180685/180691.

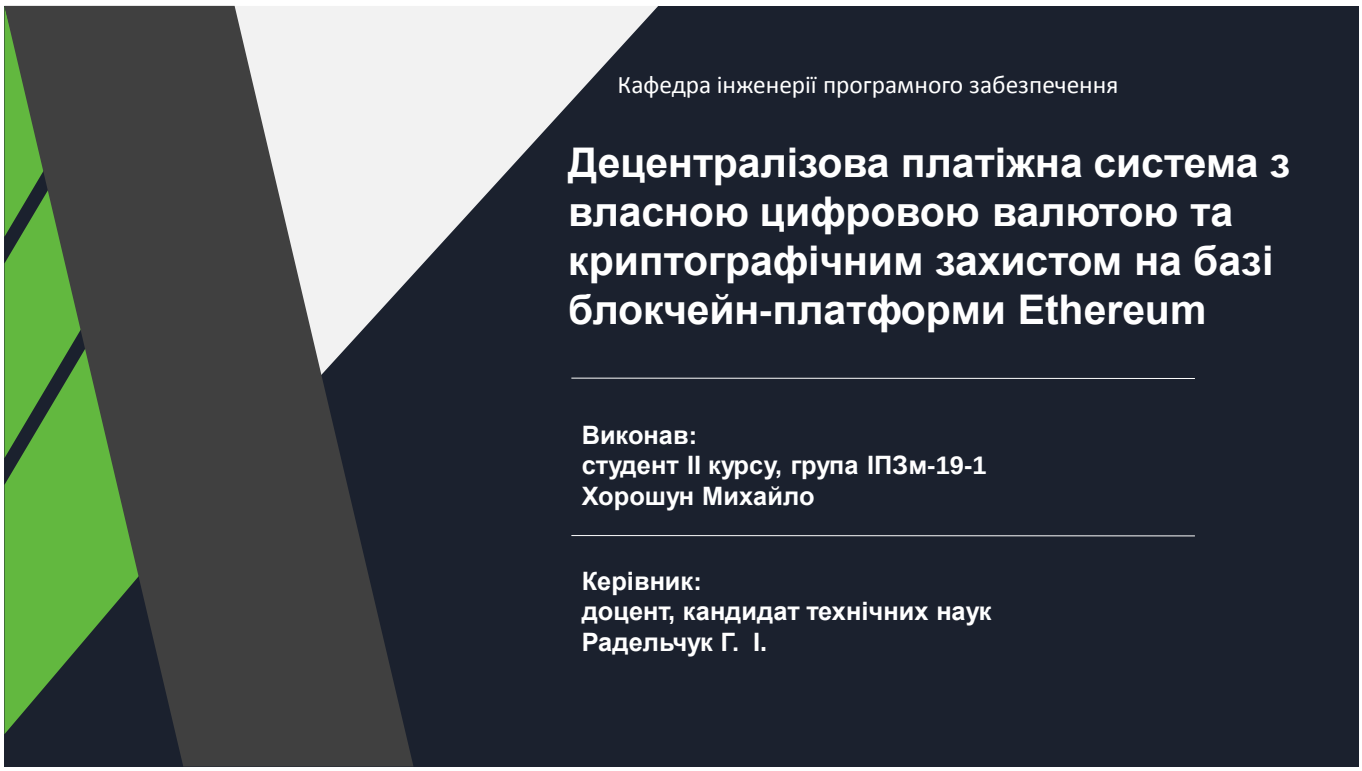
3. Schneider N. Code your own utopia: Meet Ethereum, bitcoin's most ambitious successor [Електронний ресурс] / N. Schneider // America Aljazeera. – Режим доступу: http://america.aljazeera.com/articles/2014/4/7/code-your-own-utopiameetethereumbitcoinasmostambitioussuccessor.html.

4. Koy P. This Is Your Company on Blockchain [Електронний ресурс] / P. Koy, O. Kharif / Bloomberg. – Режим доступу: https://www.bloomberg.com/news/articles/2016-08-25/this-is-your-company-on-blockchain.

ДОДАТОК Е

(обов’язковий)

ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ



Об'єкт, предмет і мета дослідження

Об’єкт дослідження – процеси функціонування децентралізованих платіжних систем, програмно-технічна база, необхідна для забезпечення обігу цифрової валюти в рамках децентралізованої платіжної системи.

Предмет дослідження – моделі та механізми створення безпечної, прозорої та ефективної децентралізованої платіжної системи, що забезпечувала б процес обігу цифрової валюти.

Мета – проектування та імплементація оригінальної моделі децентралізованої платіжної системи з обігу власної криптовалюти, яка б вирішувала наявні у галузі проблеми ефективного функціонування.

Актуальність теми

Актуальність теми роботи полягає у необхідності розробки повнофункціональної децентралізованої платіжної системи, яка б дозволяла ефективно здійснювати фінансові перекази цифрової валюти, відповідала високим стандартам безпеки, була незалежною від банківських регуляторів.

Необхідність розробки такої програмної системи ґрунтується на наявності невирішених питань в галузі криптовалютних інвестицій.

Завдання дослідження

Завданнями роботи є:
– проаналізувати специфіку функціонування ДПС;
– дослідити процедури залучення криптовалютних інвестицій та обґрунтувати необхідність розробки системи для обігу криптовалют;
– провести аналіз існуючих моделей та методів у галузі криптовалютних інвестицій, виділити невирішені проблеми;
– визначити основні вимоги до ДПС та функції, які вона має виконувати;
– удосконалити моделі та методи організації процесу функціонування ДПС, які б вирішували наявні проблеми у галузі криптовалютних інвестицій;
– виконати проектування ПС на основі розроблених методів та алгоритмів;
– виконати програмну реалізацію прийнятих рішень;
– провести тестування та практичну апробацію отриманих результатів;
– проаналізувати отримані результати та сформувати рекомендації щодо доцільності впровадження результатів дослідження.

Аналіз стану проблеми і інших рішень

Основні проблеми у галузі:

– необхідність залучення третьої сторони (криптовалютних бірж) для купівлі/продажу власної криптовалюти (Bitcoin, Ethereum).

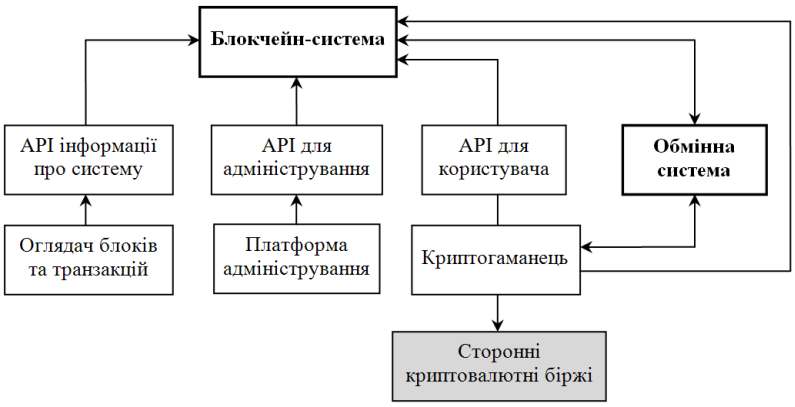
– низька швидкість проходження транзакцій (Bitcoin, Ethereum);

– функціонування системи на алгоритмі консенсусу Proof-of-Work, що вимагає значних енергозатрат (Ethereum 1.0, Bitcoin);

– недосконалий механізм розподілення винагороди між підтримувачами консенсусу.

Розроблені методи

Одним з недоліків існуючих ДПС є необхідність залучення третьої сторони для купівлі/продажу власної криптовалюти. Цю проблему було вирішено шляхом розробки власної обмінної системи.



Архітектура платформи з урахуванням власної обмінної системи

Розроблені методи

Ще одним недоліком існуючих програмних рішень є низька швидкість проходження транзакцій.

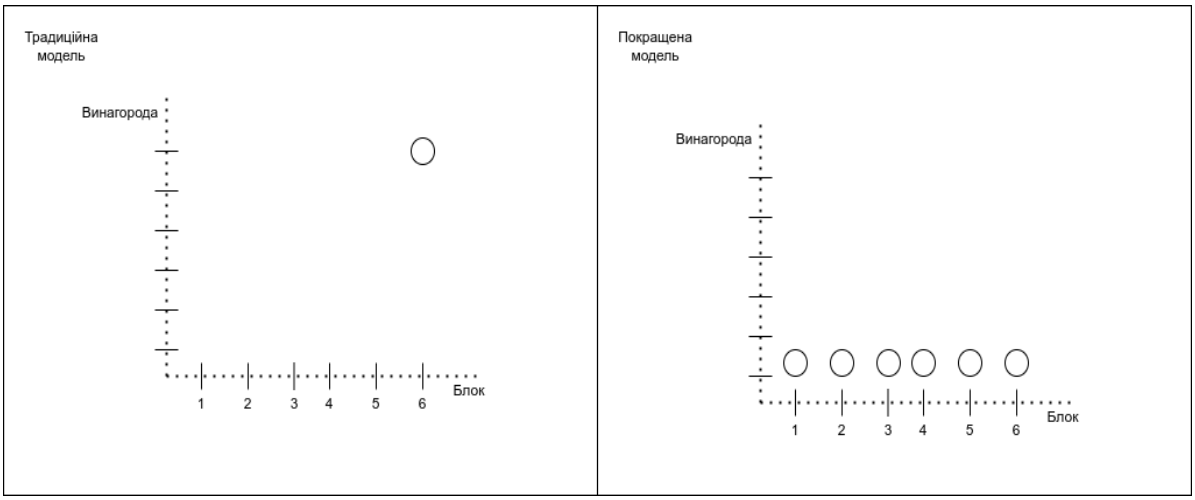
Для збільшення пропускної здатності системи було розроблено наступні методи оптимізації:

– використання динамічного розміру блоку;

– реалізація переказів «один до багатьох».

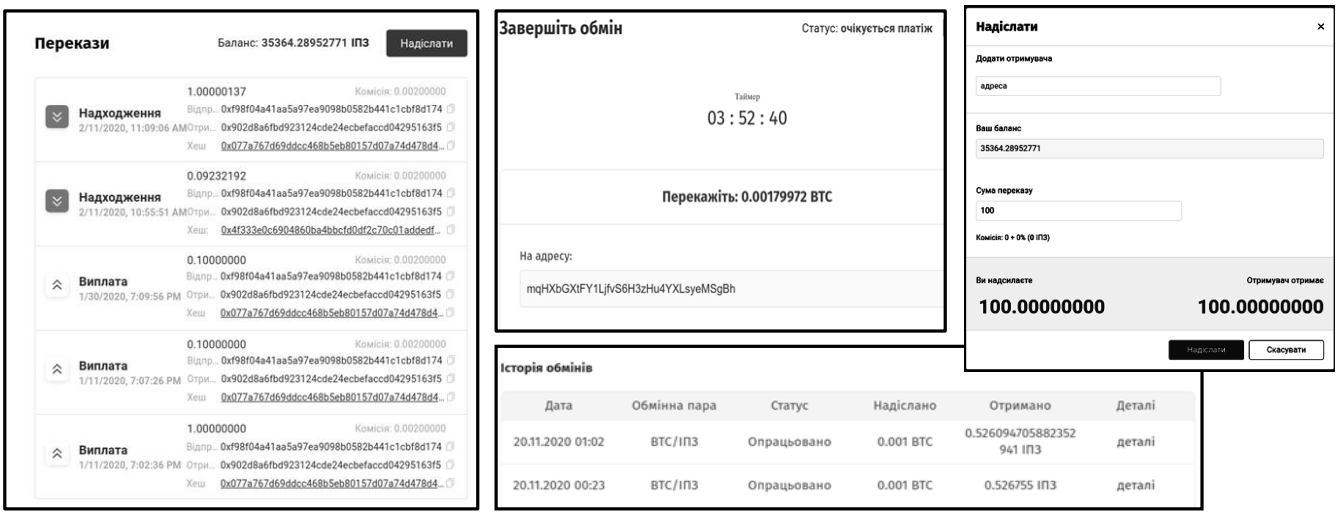
Розроблені методи

Запропоновано розподілення винагороди за кожний блок рівномірно між усіма учасниками, що брали участь у його створенні, відповідно до їхнього вкладу у підтримання консенсусу мережі.



Практична імплементація

На основі розглянутих методів поліпшення технічних характеристик системи було послідовно спроектовано та імплементовано кінцевий програмний продукт.



Отримані результати

Розроблена програмна система забезпечує:

– динамічне зростання пропускної здатність мережі при збільшенні навантаження;

– відсутність необхідності залучення третьої сторони;

– високий рівень децентралізованості завдяки використанню публічного блокчейну;

– відсутність енергозатрат для підтримки функціонування консенсусу системи;

– рівномірне розподілення нагороди між учасниками підтримання консенсусу.

Наукова новизна

1 Удосконалено метод створення цифрових платіжних засобів шляхом розробки комплексного рішення, яке складається з блокчейн-платформи та системи розумних контрактів.

2 Удосконалено метод виходу криптовалюти на відкритий ринок шляхом розробки власної криптовалютної електронної біржі.

3. Удосконалено метод знаходження консенсусу мережі, що дозволить збільшити її пропускну здатність та зменшити витрати електроенергії на її утримання.

4. Удосконалено метод розподілення винагороди за підтримання консенсусу мережі.

Практичне значення

Практична цінність отриманих результатів полягає в успішній розробці моделей та механізмів забезпечення безпечного, прозорого та ефективного процесу створення власної криптовалюти. Завдяки поліпшеним характеристикам, у порівняннні з традиційними рішеннями, розроблена програмна система має високі конкуретні шанси на ринку.

Наукові публікації

1 Опублікована одна стаття у збірнику матеріалів Міжнародної науково-практичної Інтернет-конференції:

Радельчук Г. І., Хорошун М. Л. Проектування програмної системи для залучення криптовалютних інвестицій на базі блокчейн-платформи ETHEREUM: концептуальні засади // Materiály XVI Mezinárodní vědecko-praktická konference «Vědecký průmysl evropského kontinentu – 2020», Volume 6 : Praha. Publishing House «Education and Science». – S.75–82.

2 Опублікована одна стаття у фаховому науковому виданні «Вісник Хмельницького національного університету. Серія «Технічні науки»:

Радельчук Г. І., Хорошун М. Л. Концепції проектування децентралізованої платіжної системи з власною цифровою валютою на базі блокчейн-платформи Ethereum // Вісник Хмельницького національного університету. Серія «Технічні науки». – 2020. – № 4 (287), Т. 1. – С. 89–93.

Висновки та рекомендації

В результаті виконання дипломної роботи було проведено системний аналіз у галузі криптовалютних інвестицій, визначено недоліки існуючих рішень. Запропоновано методи та рішення, які дозволять оптимізувати роботу децентралізованих платіжних систем, та доведено їх ефективність. На основі отриманих даних спроектовано та втілено у конкретний прикладний результат (програмну систему) концепцію повнофункціональної децентралізованої платіжної системи, яка дозволяє автоматизувати процеси залучення криптовалютних інвестицій.

Результати практичної апробації програмної системи підтверджують її працездатність. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у притоках інвестиційного капіталу у формі криптовалют.

Mon Dec 07 09:31:23 EET 2020, Хіврич Володимир Русланович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 3.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 14%**

	Документ		Сумарний збіг по Базі Даних	
ID: 82611 Назва: Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum Додано в БД: 2020-12-07 Автора: М.Л. Хорошун Керівники: Г. І.Радельчук Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	143222	1232	7980 (6%)	110 (9%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK

Ім'я користувача:
Кафедра ІПЗ

ID перевірки:
1005385596

Дата перевірки:
07.12.2020 11:43:46 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
07.12.2020 11:55:43 EET

ID користувача:
100005589

Назва документа: **ДР Хорошун**

Кількість сторінок: **140** Кількість слів: **21081** Кількість символів: **173155** Розмір файлу: **7.25 MB** ID файлу: **1005677564**

4.54%
Схожість

Найбільша схожість: **1.42%** з джерелом з Бібліотеки (ID файлу: **1005663221**)

3.46% Джерела з Інтернету	272	Сторінка 142
1.57% Джерела з Бібліотеки	30	Сторінка 144

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	13

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ ПО КАФЕДРІ **ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: **Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum**

Автор: **Хорошун Михайло Леонтійович**

Спеціальність: **121 «Інженерія програмного забезпечення»**

Освітня програма: **Інженерія програмного забезпечення**

Науковий керівник: **канд. техн. наук, доцент Радельчук Г. І.**

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені у роботі, є законними і не є плагіатом Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження: Текст є оригінальним, виявлені запозичення не є плагіатом, оскільки розміщені у розділах, які не описують безпосередньо авторське дослідження, складають 4,54 % та мають посилання на літературні джерела. Робота приймається до захисту.

07.12.2020 р.
Дата

Підписи

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНУ РОБОТУ

освітнього ступеня «магістр»

Магістр Хорошун Михайло Леонтійович

Тема Децентралізована платіжна система з власною цифровою валютою та криптографічним захистом на базі блокчейн-платформи Ethereum

Спеціальність 121 « Інженерія програмного забезпечення»

Обсяг дипломної роботи:

кількість листів креслень _____; кількість сторінок записки 90

1. Короткий зміст роботи та прийнятих рішень У магістерській роботі проведено аналіз процесів функціонування децентралізованих платіжних систем та програмно-технічної бази, необхідної для забезпечення обігу цифрової валюти в рамках децентралізованої платіжної системи (ДПС). Покращено моделі та механізми створення безпечної, прозорої та ефективної ДПС, яка забезпечувала б процес обігу цифрової валюти, відповідала високим стандартам безпеки та була незалежною від банківських регуляторів. На основі розроблених моделей та механізмів імплементовано програмний комплекс для ДПС з обігу власної криптовалюти, проведено практичну апробацію та доведено його ефективність.

2. Висновок про відповідність роботи дипломному завданню Дипломна робота освітнього ступеня «магістр» у повній мірі відповідає поставленому завданню як у теоретичній, так і в практичній її частині

2. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи У вступі обгрунтовується актуальність теми роботи, формулюються цілі і завдання дослідження, описується наукова новизна та практична значимість отриманих результатів. У першому розділі охарактеризовано структуру предметної області та існуючі функціональні моделі і методи організації ДПС, виконана розгорнута постановка задачі. У другому розділі досліджено методи і способи вирішення поставлених задач. Традиційні моделі системи, досліджені на етапі аналізу, були розширені новими методами та засобами з метою покращення базових характеристик системи. У третьому розділі обгрунтовано проектні рішення, що дають змогу реалізувати технічні вимоги, забезпечити сумісність та взаємодію різних компонентів системи. У четвертому розділі розглянуто питання, що стосуються реалізації програмної системи на основі прийнятих проектних рішень, а також її технічні та технологічні характеристики. Також проведено емпіричне дослідження, спрямоване на доведення працездатності розробленої системи та її функціональної придатності. Обгрунтована ефективність розроблених методів та засобів та створено рекомендації з їх застосування при проектуванні ДПС.

4. Позитивні сторони роботи Дипломна робота містить низку інноваційних рішень, зокрема, було доведено доцільність проектування власної обмінної платформи, поряд з інтеграцією зовнішніх криптовалютних бірж, для можливості купівлі розробленої криптовалюти за інші популярні цифрові валюти, покращено методику опрацювання транзакцій, що дозволило оптимізувати пропускну здатність мережі, та покращено механізм емісії винагороди за створення блоку для учасників підтримання консенсусу, забезпечивши кращу рівномірність її розподілу.

5. Негативні сторони роботи У роботі розглянуто відразу декілька способів покращення традиційної моделі ДПС за показниками різних технічних категорій, через що виникає питання: можливо краще було б дослідити лише один-два методи у вузькому діапазоні досліджуваних характеристик, але зробити це глибше і детальніше, ніж розфокусовувати дослідження до розв'язання більш широкого спектру проблем.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення виконане відповідно до теми дипломної роботи з дотриманням вимог стандартів. У загальному графічне оформлення виконане на достатньому рівні. Пояснювальна записка відповідає вимогам стандартів до її оформлення.

7. Відгук про роботу в цілому В цілому дипломна робота заслуговує позитивної оцінки. Весь матеріал дипломної роботи структурований, чіткий та послідовний. Усі розділи роботи є послідовними та логічними, що дозволяє чітко розуміти викладений матеріал у рамках тематики дипломної роботи. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті за основу для вирішення поставленої задачі.

8.Інші зауваження

9. Оцінка дипломної роботи Розглянувши позитивні та негативні сторони представленої дипломної роботи, можна зробити висновок, що вона заслуговує оцінки «відмінно».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Мартинюк Валерій Володимирович, професор, завідувач кафедри «АКІТ» (Автоматизації і комп'ютерно-інтегрованих технологій) ХНУ

« 1 » 12 2020 р. _____
(підпис)