Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2022 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»
освітньо-професійна програма «Мережеві та інтернет технологій»

на тему:

ЗАБЕЗПЕЧЕННЯ ДОСТУПУ ДО СОЦІАЛЬНИХ ПОСЛУГ З ВИКОРИСТАННЯМ ТОКЕНІВ

Виконав: студент групи МІТ -41

Віктор ІЛЬНИЦЬКИЙ _____
(прізвище ім'я по-батькові) (підпис)

Керівник: доцент кафедри мережевих та інтернет технологій

к.т.н., Оксана ГЕРАСИМЕНКО _____
(посада, прізвище ім'я по-батькові) (підпис)

Київ 2022

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій

_____ Ю.В. Кравченко

«_____»_____2022 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Ільницький Віктор Вікторович
(прізвище, ім'я, по батькові)

1. Тема роботи: Забезпечення доступу до соціальних послуг з використанням токенів
затверджена на засіданні кафедри МІТ «24» грудня 2022 р. протокол № 8
2. Термін здачі закінченої роботи «30» травня 2022 р.
3. Вихідні дані до проекту (роботи)
Типовий контракт для створення токенів на Solidity
4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)
Вступ
1. Аналіз проблем та перспектив використання технології блокчейн у керуванні доступом до послуг
2. Розробка концепції керування доступом до соціальних послуг з використанням токенів
3. Реалізація смарт-контрактів
Висновки

5. Перелік графічного матеріалу, слайдів
1. Постановка задачі 2. Токени як засіб доступу до послуг
3. Загальна концепція доступу до соціальних послуг з використанням токенів
4. Схема взаємодії контрактів 5. Засоби реалізації
6. Смарт-контракти 7. Розгортання додатку 8. Висновки

Дата видачі завдання _____
Керівник роботи _____ доц. Оксана ГЕРАСИМЕНКО
(підпис) (посада, прізвище, ім'я, по батькові)

Завдання прийняла до виконання _____ Віктор ІЛЬНИЦЬКИЙ

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	01.02.2022	
2	Розділ 1	01.03.2022	
3	Розділ 2	01.04.2022	
4	Розділ 3	15.05.2022	
5	Доповідь та слайди	30.05.2022	
6	Пояснювальна записка	30.05.2022	

Здобувач вищої освіти Віктор ІЛЬНИЦЬКИЙ

Керівник Оксана ГЕРАСИМЕНКО

РЕФЕРАТ

Пояснювальна записка: 37 с., 16 рис., 18 джерел.

Мета роботи(проекту): розробити концепцію надання соціальних послуг на основі технології блокчейн з використанням токенів; на прикладі блокчейн Ethereum розробити схему взаємодії контрактів для реалізації запропонованої концепції.

Об'єкт дослідження: наявна система соціального захисту населення.

Предмет дослідження: процес забезпечення доступу до соціальних послуг з використанням токенів.

Методи дослідження: системний аналіз.

Короткий зміст роботи: досліджено сучасну систему забезпечення соціальних послуг в Україні, показано основні проблеми, що є у даній сфері, простежено взаємозв'язки між тими чи іншими факторами, що впливають на предмет дослідження, виявлені закономірності, що призводять до негативних наслідків в системі, обґрунтовано доцільність проведеної роботи. Результати здійснених у дипломному проекті досліджень можуть бути використані в подальшій модернізації системи надання соціальної допомоги в Україні. Наукова новизна дослідження – запропоновано концепцію надання соціальних послуг на основі технології блокчейн з використанням токенів.

Ключові слова: ТЕХНОЛОГІЯ БЛОКЧЕЙН, ТОКЕН, КРИПТОВАЛЮТА, СМАРТ-КОНТРАКТ, СОЦІАЛЬНА ДОПОМОГА, АВТОМАТИЗАЦІЯ

ЗМІСТ

ВСТУП

Актуальність дослідження. Проблема соціального захисту населення в Україні була завжди актуальна. Значні економічні та політичні процеси, які відбулись в нашій країні протягом останніх років, привели до того, що з'явилися нові категорії населення, які потребують допомоги від держави, наприклад, учасники бойових дій, переселенці тощо. З іншого боку сама система надання соціального захисту далеко не ідеальна і має багато проблем. Це насамперед непрозорість даного процесу, що тягне за собою зловживання посадовими особами ситуацією в своїх корисливих цілях і призводить до значних розкрадань і так дефіцитних ресурсів. Наприклад, отримання соціальної допомоги громадянами, які на неї не мають права і отримали її незаконно шляхом зв'язків з відповідальними за це особами. Поширеними є скарги громадян про незрозумілі маніпуляції з чергами на пільгове житло, земельні ділянки тощо. Також громіздка бюрократична тяганина ускладнює швидкий доступ населення до допомоги, особливо в регіонах де відбуваються бойові дії і є перебої в роботі органів державної влади. Дане дослідження покликане вирішити проблему непрозорості та бюрократичної тяганини, що дозволить максимально ефективно розподіляти уже наявні ресурси.

Ступінь розробки. У даній роботі пропонується використати новітню технологію блокчейн, яка ідеально підходить для вирішення поставлених завдань. Була розроблена система на основі смарт-контрактів із застосуванням токенів. Так як це доволі нова технологія, яка тільки починає використовуватись в різних сферах життя, аналогів використання в державному регулюванні таких процесів в світі немає. Україна давно є одним з лідерів по впровадженню цифрових технологій в управлінні, тому застосування блокчейну в розподіленні соціальної допомоги серед населення буде черговим вагомим кроком вперед. З іншого боку, для вирішення поставлених задач застосування бехнології блокчейн є визначальним, так як існуючі системи, що зберігають данні про громадян, які отримують допомогу, можуть бути скомпрометовані тими, хто до них має доступ,

тобто посадові особи можуть внести зміни в своїх корисливих цілях і це майже неможливо відслідкувати через непрозорість системи.

Практичне значення одержаних результатів. Дане дослідження показало, що розроблена система максимально ефективно автоматизує надання соціальної допомоги, зменшуючи бюрократичний тягар тим самим економлячи час як громадян, так і посадових осіб, що розвантажує державний апарат. Також дана розробка робить весь процес прозорим, що дозволяє контролювати адресність допомоги та унеможливлює махінації. У той же час запропоноване рішення дозволяє автоматизувати оплату надавачам послуг без стороннього втручання.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Blockchain – база даних що зберігає деякий ланцюг записів та шифрує їх[1]

Смарт-контракт – договір, що виконується при його викликанні, умови якого записуються безпосередньо в блокчейн.

Solidity – об’єктно-орієнтована мова програмування, розроблена для написання смарт-контрактів.

Ethereum – децентралізована система, розроблена спеціально для взаємодії з смарт-контрактами, що використовує власну криптовалюту – ефір.

Криптовалюта – це новий вид активів, який відрізняється від звичайних, фіатних грошей тим що для транзакцій та володіння нею не потрібні послуги банку чи якийсь контролюючий орган.[2]

Фіатні гроші – тип грошей або валюти, цінність яких походить не від власної вартості або гарантії обміну на золото або іншу валюту, а від державного наказу (fiat) використання їх як засобу платежу.[3]

DApps - децентралізований додаток.

Інтернет речей (IoT) – концепція мережі, що складається із взаємозв'язаних фізичних пристроїв, які мають вбудовані давачі, а також програмне забезпечення, що дозволяє здійснювати передачу і обмін даними між фізичним світом і комп'ютерними системами в автоматичному режимі, за допомогою використання стандартних протоколів зв'язку.[4]

1 АНАЛІЗ ПРОБЛЕМ ТА ПЕРСПЕКТИВ ВИКОРИСТАННЯ ТЕХНОЛОГІЇ БЛОКЧЕЙН У КЕРУВАННІ ДОСТУПОМ ДО ПОСЛУГ

1.1 Проблеми надання соціальної допомоги в Україні

Соціальна допомога громадянам є одним з основних завдань кожної держави. Складна бюрократична процедура отримання допомоги та високий рівень розкрадань в Україні приводять до зниження якості надання соціальних послуг. Це веде до збільшення недовіри малозабезпеченими групами населення державі та владі. Тому потрібно удосконалювати систему розподілення соціальними благами задля мінімізації розкрадань та нечесного розподілу допомоги.

Соціальна допомога – це широкий комплекс заходів, призначений захистити та покращити матеріальний стан груп населення, які того потребують найбільше, шляхом створення низки соціальних програм та забезпечення відповідних соціальних закладів. Допомога може надаватися як в грошовій, так і в натуральній формі (безоплатні товари, оздоровчі заходи тощо), фінансується з державного бюджету та пожертвувань благодійних організацій та надається по певному визначеному принципу (вік, дохід, кількість дітей тощо).

Основним принципом такої допомоги є адресність. Адресність це надання соціального захисту саме тим громадянам, які того найбільше потребують з урахуванням їхнього матеріального стану та місячного доходу, що строго перевіряється наданням низки відповідних документів. Тобто суть адресності – це ефективний розподіл певних ресурсів між тими, хто найбільше потребує задля підвищення їх матеріального становища при мінімальних затратах від держави, що дозволяє покрити чим більше громадян. У той же час адресність дає можливість пропорційно розподілити ресурси, збільшивши частку найбільш нужденних, при цьому ефективно використавши бюджетні кошти.

На жаль, в нашій країні є проблема заполітизованості соціальної допомоги, що порушує баланс розподілу коштів та створює непорозуміння в суспільстві.

Брак єдиної думки щодо реформування даних процесів тільки посилює проблему і відтягує реформування всієї системи. Також оновленню існуючого механізму перешкоджає дороговартісна зміна адміністративного устрою по розподіленню соціальної допомоги. Якщо порівняти адресну та універсальну допомогу, то остання призначається широкій групі населення без чітких характеристик громадянина. Перевагою адресної допомоги є те, що вона призначається конкретній групі малозабезпеченого населення і дозволяє ефективно допомогти тим, хто цього потребує найбільше, в той час як пільги призначаються більш широкій по матеріальному становищу групі громадян і не призводять до справедливого балансу.

Наступною проблемою надання соціальних послуг є бюрократична тяганина. Ця проблема існувала завжди і вона притаманна не тільки Україні. Постійна зміна законів та положень щодо переліку потрібних документів для отримання допомоги та інші фактори ускладнюють доступ до соціального забезпечення як в часі, так і в деяких випадках лишають доступу зовсім. Наглядним прикладом є бойові дії в деяких регіонах нашої держави, що призвели до проблем з доступом до адміністративних ресурсів та частково унеможливило отримання соціального захисту. На щастя, наша держава займає передову позицію в впровадженні цифрових технологій. З появою застосунку «Дія» громадяни отримали доступ до своїх документів в цифровому вигляді та можливість з телефону отримати ті чи інші соціальні послуги, не взаємодіючи з бюрократичним апаратом. Проте не всі громадяни є обізнаними в цифрових технологіях або ж зовсім не мають можливості отримати доступ до даного ресурсу, особливо ті, які того потребують найбільше. Не дивлячись на це, впровадження таких технологій на рівні держави є позитивним фактором і з часом дана екосистема стане ще розвиненішою та кожен зможе нею користуватися.

Наступною проблемою доступу до соціальної допомоги є непрозорість системи та корупція, що випливає з цього. Дані про надання тих чи інших послуг в більшості своїй це закрита інформація, доступ до якої мають тільки певні посадові особи і керуючі органи. До пересічних громадян доходить тільки

загальна інформація про виділені кошти на ту чи іншу сферу, а далі прослідкувати, як вони розподіляються, дуже важко і затратно. Це створює прецедент для зловживань людьми, що мають доступ до цих грошей чи ресурсів. У людей природньо виникає недовіра до влади і системи, а це негативно позначається на іміджі держави.

1.2 Використання технології блокчейн у сфері надання послуг

1.2.1 Короткий огляд технології блокчейн

Блокчейн доволі нова технологія, яка набрала популярність через криптовалюти – це новий вид активів, який відрізняється від звичайних, фіатних грошей тим, що для транзакцій та володіння нею не потрібні послуги банку чи якийсь контролюючий орган.

Схема, за якою працює блокчейн, наведена на (рис. 1.1). Блок записує певну кількість транзакцій, після чого дана інформація зашифровується в хеш-функцію. Потім створюється новий блок, який містить в собі хеш попереднього плюс таку ж кількість нових транзакцій і знову все зашифровується. Цей процес циклічний і відбувається по ланцюжку.

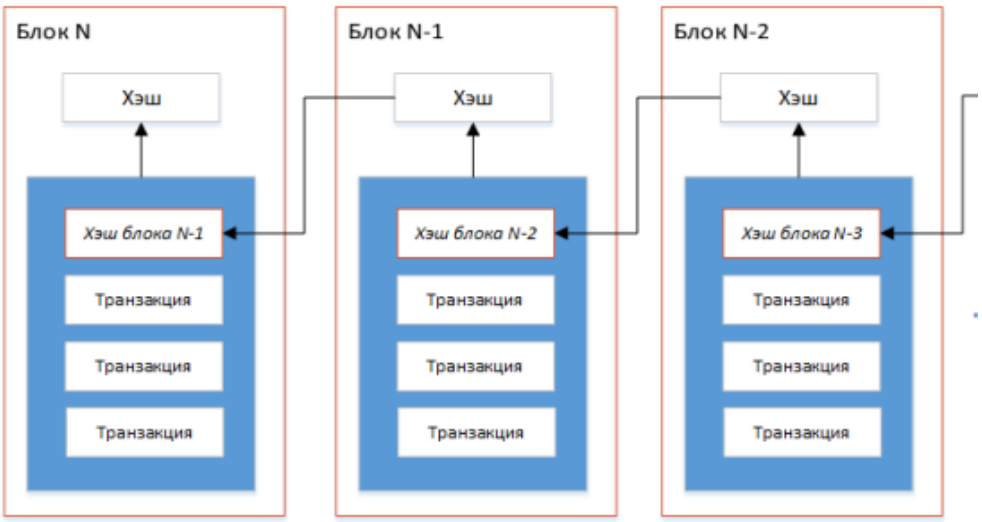

Рисунок 1.1 – Схематичне представлення ланцюжка блоків у блокчейн

Таким чином, змінити інформацію у вже сформованих блоках неможливо, тому що при мінімальній зміні даних блоку хеш-функція буде повністю інша. Ця система є децентралізованою, тобто користувач, який має потужності для обчислення, формує блок, після чого він звіряється з блоками, які сформували інші користувачі та, якщо вони співпадають, починається формуватися новий блок. Користувачі, які беруть в цьому участь, отримують винагороду у вигляді комісії. Таким чином, не потрібен жоден контролюючий орган-посередник.

Шифрування блоків відбувається за допомогою алгоритмів хешування.Хеш – це число, яке генерується з тексту за допомогою хеш-алгоритму, що показано на рисунку . Це число менше оригінального тексту. Алгоритм працює так, що для кожного тексту генерується унікальний хеш. І відновити текст із хеша, перехопивши повідомлення, практично неможливо. Хешування потрібне, щоб інформація у базах даних не дублювалася, для цифрових підписів та SSL-сертифікатів, щоб знайти конкретну інформацію у великих базах даних, у комп'ютерній графіці та в технології блокчейн і в багатьох інших випадках.

INPUT	HASH
Hi	3639EFCD08ABB273B1619E82E78C29A7DF02C1051B1820E99FC395DCAA3326B8
Welcome to blockgeeks. Glad to have you here.	53A53FC9E2A03F9B6E66D84BA701574CD9CF5F01FB498C41731881BCDC68A7C8

Рисунок 1.2 – Приклад хешування тексту

Одна з незамінних властивостей хешування – його унікальність. Те саме значення хеша не може використовуватися для різного тексту. Найменша зміна тексту повністю змінить значення хеша. Це називається ефектом лавини і продемонстровано на рисунку 1.3

INPUT	HASH
This is a test	C7BE1ED902FB8DD4D48997C6452F5D7E509FBCDBE2808B16BCF4EDCE4C07D14E
this is a test	2E99758548972A8E8822AD47FA1017FF72F06F3FF6A016851F45C398732BC50C

Рисунок 1.3 – Демонстрація ефекту лавини

Проте, в цієї технології куди більший потенціал застосування, ніж просто віртуальні гроші. Вона знайшла застосування у різних сферах, таких як розробка ігор, зберігання та ідентифікація персональних даних, контроль за поставками товарів тощо.

1.2.2 Смарт-контракти та проблеми їх створення

Смарт-контракт – комп'ютерний код, який виконує угоду між двома та більше сторонами, в результаті чого відбуваються певні дії, прописані в контракті. Тобто, коли задіюється запрограмована умова, смарт-контракт автоматично виконує угоду. Якщо розглядати звичайний контракт та смарт-контракт, то можна точно сказати, що вони обидва є угодами, в яких дві або більше сторони погоджуються дотримуватися низки умов. Їхні фундаментальні елементи однакові: добровільна згода всіх сторін, об'єкт договору та єдина мета. Тим не менш, обидва відрізняються за трьома факторами: спосіб написання, його юридичні наслідки та концепт дотримання.

Розумні контракти повністю цифрові та написані мовою програмування. У доповнення до встановлення зобов'язань та наслідків так само, як і у звичайному фізичному документі, код може виконуватися автоматично. Отже, він може отримувати та обробляти інформацію, вживаючи заходів відповідно до правил договору.

Однаєю з найпопулярніших платформ створення смарт-контрактів є платформа Ethereum. Вона замінює більш обмежену мову BTC мовою, що дозволяє розробникам задавати власні сценарії. Ethereum дозволяє розробникам програмувати власні смарт-контракти. Платформа Ethereum використовувалася

для поширення DApps. Замість багатьох програм, керованих багатьма протоколами, Ethereum дозволяє керувати всіма програмами по одному протоколу.

Ethereum – це платформа, яка дозволяє розробникам створювати будь-яку програму і запускати її на основних функціях блокчейна, використовуючи смарт-контракти для автоматичного виконання своїх дій, задіявши зумовлені умови, вбудовані в алгоритм. У випадку, якщо умови будуть виконані, ця функція автоматично завершується без необхідності вживання будь-яких дій.

Використовуючи смарт-контракти, більше не потрібно вдаватися до допомоги третьої сторони, наприклад, адвоката або нотаріуса, який, крім можливих помилок, тягне за собою значні витрати. Блокчейн здатний захистити інформацію в зашифрованій мережі, до якої можна звертатися з будь-якої точки світу, тому швидкість та безпека очевидні. Найголовнішими перевагами контрактів є:

- автономність – ці контракти завжди укладаються між однією чи декількома фізичними чи юридичними особами, але без посередників. На підтвердження договору не потрібен юрист. Тому сторони скорочують і можуть навіть усунути будь-яку зайву людину, яка не бере участі в договорі;
- зниження витрат – оскільки контракти не залежать від третіх сторін, витрати знижуються. Найменше втручання людини призводить до зниження витрат;
- швидкість – розумні контракти використовують програмний код для автоматизації завдань, які інакше виконувалися б вручну. Тому вони збільшують швидкість бізнес-процесів і менш схильні до помилок вручну;
- безпека – основуючи контракти на блокчейні Ethereum, вони не можуть бути втрачені. Все є незмінним. Ніщо і ніхто не може змусити його зникнути. Процес децентралізованого управління виключає ризик маніпуляцій, оскільки виконання керується автоматично всією мережею, а не окремою її частиною.

З іншого боку, програми такого типу мають певні недоліки. Головний полягає переважно у використанні технологій, які він залучає: інтернет-речей та блокчейн. IoT може забезпечити зв'язок з активами, але він ще має пройти довгий шлях у сфері безпеки. IoT-пристрої легко зламати. Блокчейн більш ніж безпечний, але незмінний. Після того, як умови узгоджені, їх не можна буде згодом змінити, що може бути невигідно для когось із сторін.

1.2.3 Автоматизації процесу доступу до соціальних послуг з використанням смарт-контрактів

Одним із завдань даного дослідження є автоматизація процесу доступу до соціальних послуг з використанням смарт-контрактів. Завдяки цій технології можливо максимально автоматизувати систему надання допомоги, а всю бюрократичну процедуру звести до мінімуму. У даному дослідженні надання тому чи іншому громадянинові доступу до соціальних послуг реалізовано через певні політики, які визначаються на етапі реєстрації організації, яка представляє ту чи іншу послугу. Проте в майбутньому, якщо подібна система буде реалізована в нашій державі, можна використати документи та інтерфейс застосунку «Дія». Система оброблятиме запит громадянина, використовуючи його документи, тим самим перевіряючи, чи підходить він під критерії на отримання допомоги, після чого буде або підтверджено, або відхилено запит громадянина. У разі успіху проходження перевірки політики, користувачу надходитиме певна кількість токенів для оплати за зазначену послугу. Таким чином, цей процес являється повністю автоматичним та не потребує участі третьої сторони.

1.3 Аналіз наявних технологій керування доступом до послуг з використанням блокчейн

Так як уже було зазначено, Україна є передовою державою в цифрових технологіях, тому блокчейн уже декілька років використовується для надання

певних послуг у країні. У 2017 році було реалізовано оновлену інформаційну систему державного земельного кадастру. Впровадження даної системи дозволило забезпечити надійну синхронізацію даних, що унеможливило їх підміну в результаті зовнішнього втручання, а також надало доступ суспільству до моніторингу змін.

Також у 2017 році була реалізована ще одна система за допомогою даної технології «СЕТАМ» – системи електронних торгів арештованим майном. Був проведений перший в світі аукціон за допомогою цієї технології, що унеможливлює зміни даних у системі. Для реалізації блокчейну була обрана платформа EXONUM міжнародної компанії BitFury Group. За час роботи аукціону було продано 36 599 лотів на 10,4 млрд грн.

1.4 Постановка задачі

Як показує практика, технологія блокчейн уже знайшла своє застосування та показала ефективність в сфері послуг в Україні. Тому реалізація системи для надання соціальних послуг за допомогою блокчейну є актуальною. Потрібно розробити автоматизовану систему, зручну й зрозумілу для кінцевого користувача, яка повинна реєструвати організації, які хочуть надавати свої послуги, встановлюючи їм певні політики, котрі будуть максимально ефективно розприділяти послуги серед громадян, які потребують їх найбільше. Також повинна бути розроблена чітка та швидка система верифікації користувача за наявними політиками та розподіл токенів. У той же час сервіс повинен забезпечити договір між громадянином та організацією, що надає послуги, та передати токени після успішного завершення угоди.

Висновки по розділу 1

Отже, проаналізувавши проблеми та перспективи використання технології блокчейн в забезпечені доступу до соціальних послуг, було показно актуальність

даного дослідження. Застосування технології вирішить основні наявні проблеми при існуючій системі розподілення, проте так як технологія нова, потрібен буде час для сприйняття суспільством.

2 РОЗРОБКА КОНЦЕПЦІЇ КЕРУВАННЯ ДОСТУПОМ ДО СОЦІАЛЬНИХ ПОСЛУГ З ВИКОРИСТАННЯМ ТОКЕНІВ

2.1 Токени як засіб доступу до послуг

Токен – це цифровий сертифікат, який гарантує зобов'язання компанії перед його власником, аналог акцій на фондовій біржі у світі криптовалют. У віртуальному світі токен є цифровою умовною одиницею, вартість якої виражається в будь-якому активі. Він синхронізований з базою даних, побудованою на технології блокчейн, де ведеться підрахунок усіх токенів. Отримати доступ до віртуальних токенів можна лише за наявності електронного підпису та через відповідну програму.

Звичайно, на даний момент не існує якоїсь єдиної класифікації, проте на сьогоднішній день токени можна розділити на такі види:

- Security tokens (інвестиційні) – створені для спрощення роботи інвесторів і є аналогічними до акцій компанії. Вони засвідчують право власності та дають змогу отримувати дивіденди;
- Utility tokens (службові) – призначені для створення віртуальної валюти всередині бізнесу, компанії, будь-якої платформи. Зазвичай службові токени виражають бали, одержувані у виконанні акцій підприємств, також до них відносяться ігрові валюти тощо;
- Asset-backed tokens (сировинні) – це токени, забезпечені реально існуючими ліквідними активами. Це можуть бути як товари та послуги, так і нафта із золотом. Компанія, що випускає сировинні токени, зобов'язана виплатити власнику вартість токена або надіслати товар в обмін на токени.

Як відомо, управління криптовалютою здійснюється децентралізовано. Простіше кажучи, за функціонування монет відповідає певний алгоритм, який не можна регулювати. Токени ж, на відміну криптовалюти, можуть бути як децентралізованою валютою, так і централізованою. В останньому випадку за управління монетою відповідає єдина компанія, яка і є її творцем. У цій же

організації відбуваються всі угоди, проводяться транзакції та опрацьовується вся інформація, пов'язана з обліком монет.

2.2 Загальна концепція доступу до соціальних послуг з використанням токенів

Дане дослідження розглядає систему надання послуг особам, які належать до соціально незахищених верств населення шляхом безоплатного доступу до певних послуг. Надання прямої фінансової допомоги у рамках даної роботи не розглядається.

При розробці концепції виходимо з того, що базова платформа розгортання смарт-контрактів – платформа Ethereum. Для роботи з цією платформою, в тому числі і для розгортання смарт-контракту, як фізична, так і юридична особа, повинна мати власний рахунок.

Керування рахунком здійснюється з акаунта криптогаманця. У якості криптогаманця можна скористатися MetaMask. Це спеціальний додаток, який можна встановити як окремо, так і у вигляді розширення до браузера Chrome. Наприклад, на рисунку 2.1 показано акаунт MetaMask, у якого на рахунку є 100 ETH. Таким чином, вважаємо, що кожна фізична особа та юридична організація, які є учасниками сфери соціального обслуговування, мають свій рахунок на блокчейн-платформі Ethereum.

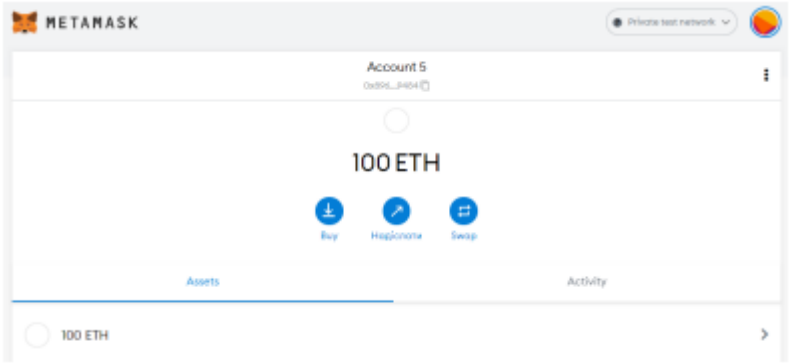

Рисунок 2.1 – Акаунт в MetaMask

Умовно назвемо організацію, яка від імені державної влади здійснює адміністрування надання послуг, «Центром надання послуг» або скорочено просто центром. Громадяни, які звертаються за отриманням соціальної допомоги, у подальшому будемо називати клієнтами центру. Центр визначає політики надання послуг, реєструє організації, які можуть надавати послуги, та проводить розрахунки з ними, веде облік клієнтів, які звернулися із запитом на отримання послуги, та інше. Варто зазначити, що описана діяльність представляється відповідними функціями смарт-контракту.

Токен у даному проекті являє собою цифровий ідентифікатор, який підтверджує дозвіл клієнта на отримання послуги. Центр у рамках своєї діяльності випускає певну кількість токенів, кожному з яких відповідає деяка кількість Wei (частина від ETH, грошової одиниці платформи Ethereum). При реєстрації клієнта, центр визначає політики доступу клієнта до послуг та перераховує необхідну кількість токенів для майбутньої оплати отриманих послуг. Присвоєння політик та перерахування токенів, як і оплата ними за послуги, програмується у смарт-контракті.

У свою чергу організація, яка надає послуги (далі сервісна організація), реєструється у системі та отримує підтвердження на здійснення свого виду діяльності. Клієнт може обрати серед сервісних організацій, які надають однакову послугу, ту, яка йому найбільше підходить, та ініціює отримання послуги. За результатами її отримання відповідний токен з рахунку клієнта передається на рахунок сервісної організації.

Сервісна організація може ініціювати передачу зароблених токенів «Центру надання послуг», у результаті чого отримає на свій рахунок відповідну їм кількість Wei.

На рисунку 2.2 схематично представлено взаємодію основних сутностей, котрі приймають участь у наданні/отриманні соціальних послуг.

Доступність тих чи інших сервісів (послуг) клієнту визначається політиками. Їх призначення – регулювати доступ клієнта лише до тих послуг, які йому доступні згідно підтверджених «Центром надання послуг» категорій.

Категорія визначає, до якої групи соціально-незахищених осіб відноситься клієнт, наприклад, багатодітна сім'я, дитина-сирота та інше. Одна особа може одночасно належати до кількох категорій.

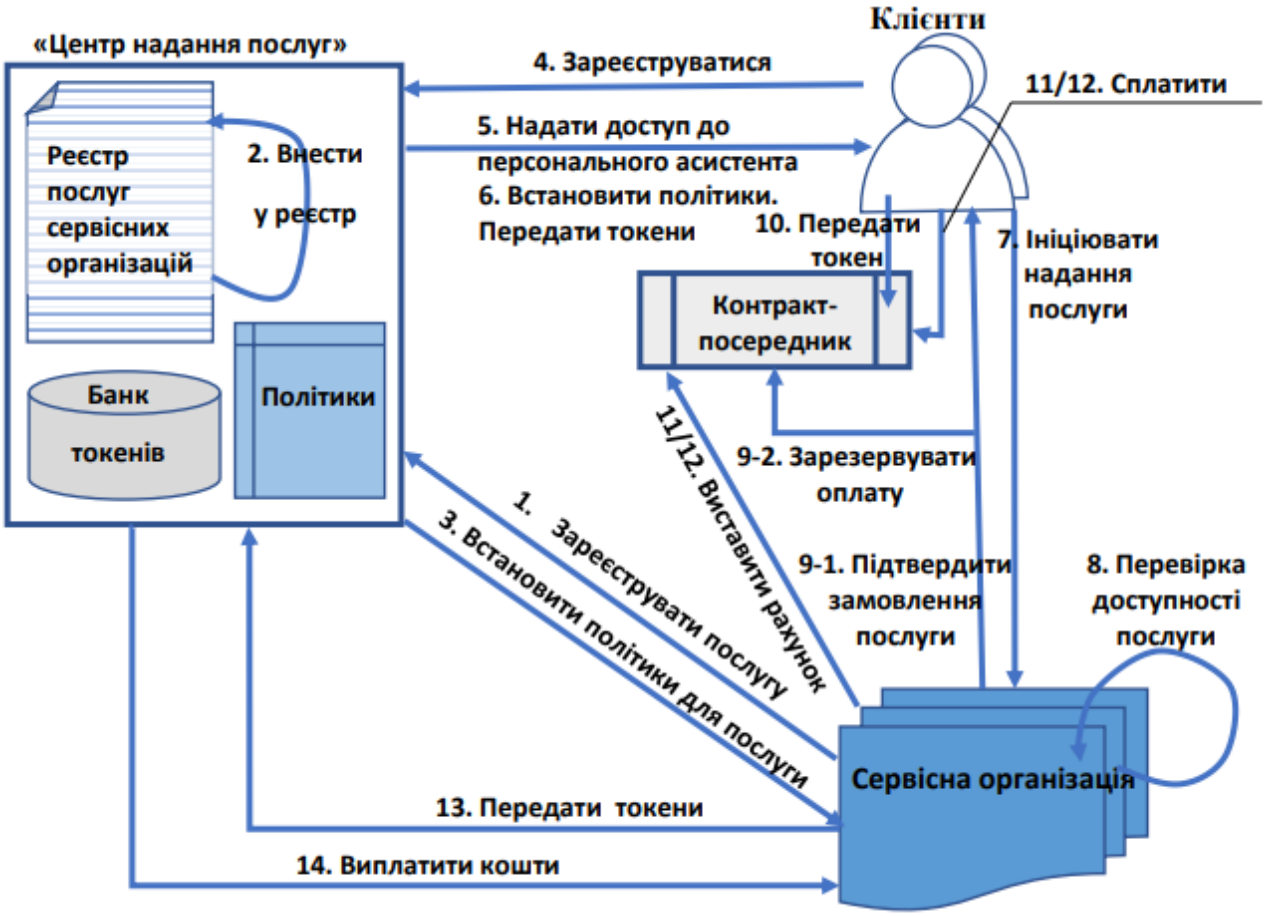

Рисунок 2.2 – Схема взаємодії учасників сфери соціального обслуговування

Політики встановлюються як клієнту, так і сервісу. Встановлення політик сервісу відбувається у момент, коли організація реєструє його у центрі. Співставлення політик, встановлених клієнту, та політик, встановлених сервісу, дозволяє визначити, чи доступно надання клієнту вибраної ним послуги.

2.3 Вибір засобів реалізації програмного додатку

2.3.1 Вибір мови програмування для створення смарт-контракту

Дана система розробляється на базі блокчейну Ethereum, який має власну мову програмування Solidity, розроблену спеціально для написання смарт-контрактів.

Solidity – об'єктно-орієнтована мова для розробки смарт-контрактів. Є кросплатформеною, але практично використовується переважно на Ethereum.

Була створена у 2014 році командою програмістів під керівництвом Крістіана Райтвізнера на основі ідеї Гевіна Вуда. Схожість синтаксису на JavaScript – розрахунок на швидку адаптацію розробників, які на ньому розробляли подібні протоколи. Solidity підтримує спадкування, зокрема множинне, зокрема з C3 лінеаризацією. Solidity — статично типізована мова, динамічними є тільки типи, що повертаються.

Підтримується бінарний інтерфейс програмування (ABI), що має безліч типобезпечних функцій у кожному контракті (згодом з'явився також і Serpent). Специфікована система документування коду для пояснення послідовності викликів, що отримала назву «Специфікації природною мовою Ethereum» (Ethereum Natural Specification Format) У Solidity замість звичних класів оголошуються контракти (contract). Існують бібліотеки для написання смарт-контрактів, такі як: Open Zeppelin, Truffle. Бібліотеки дозволяють створити свою монету (токен) на основі готових шаблонів, з усіма специфікаціями (ERC20) та перевірками на безпеку (бібліотека safemath). Контракти у solidity можуть успадковуватися один на одного. Це означає, що функції та змінні контракту, від якого виконується спадкування, будуть доступні у контракті-потомку. У Solidity, як і в C++, є множинне спадкування.

2.3.2 Вибір засобів написання, компіляції та розгортання смарт-контрактів

Перед початком роботи, перш за все, потрібно вибрати середовище програмування. Універсальним варіантом є Remix IDE, яке можна використовувати безпосередньо через браузер.

Remix – це Solidity IDE, яка використовується для написання, компіляції та налагодження коду Solidity, зображено на рисунку 2.3

IDE означає інтегроване середовище розробки і є програмою з набором інструментів, призначених для допомоги програмістам у виконанні різних завдань, пов'язаних з розробкою програмного забезпечення, таких як написання, компіляція, виконання та налагодження коду.

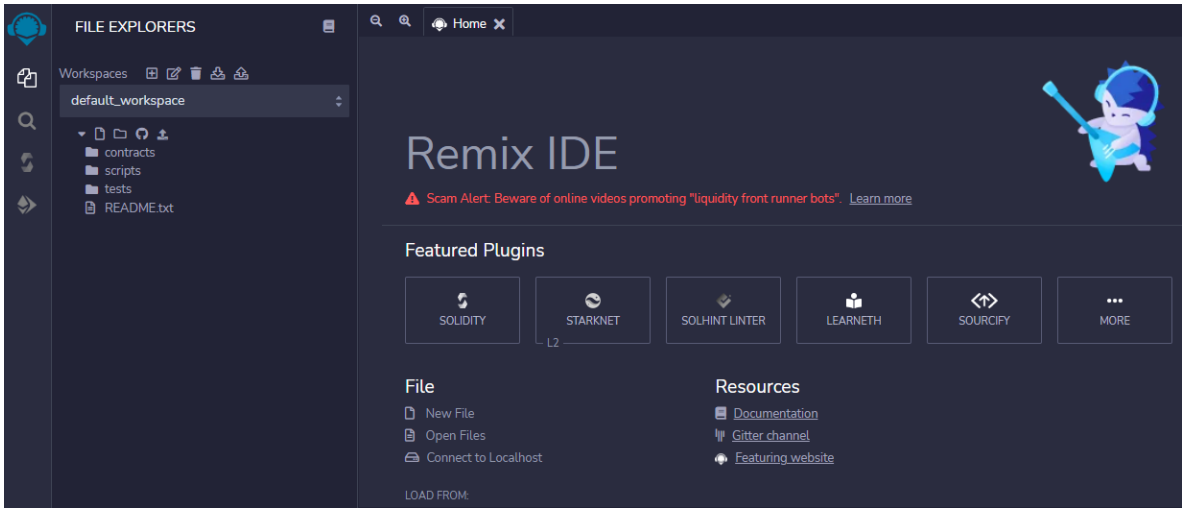

Рисунок 2.3 – Середовище програмування смарт-контрактів Remix IDE

Remix – React-фреймворк для SSR (server-side rendering). Це означає, що бекенд та фронтенд реалізуються в одному додатку. Рендер відбувається на сервері і верстка передається на клієнт із мінімальним використанням JavaScript. У класичному React-додатку дані спочатку окремо фетчаться, а потім компоненти з ними рендеряться на фронті. Remix, навпаки, отримує дані на бекенд і передає відрендерений HTML безпосередньо користувачеві.

Remix, як і інші фреймворки, має кілька особливостей "з коробки", які роблять його зручним для розробників. Будь-яка сторінка всередині папки будь-

якого роуту є вкладеним роутом, а не окремим. Тому можна вставити ці компоненти всередину батьківської сторінки, що скорочує час завантаження. Remix автоматично обробляє всі завантаження. Все, що потрібно зробити – сказати, що потрібно показувати під час завантаження програми. У фреймворках на кшталт Next.js знадобиться встановити стан завантаження за допомогою якоїсь бібліотеки керування станом, наприклад, Redux або Recoil. Тобто на інших фреймворках для цього потрібні додаткові дії, а Remix це вбудована можливість.

Написання смарт-контрактів є складним процесом, кожен раз при розгортанні блокчейн потребує комісію, до того ж неправильно реалізувавши той чи інший контракт можна й зовсім втратити реальні кошти, якщо розортати контракт в основній мережі. Тому існуює можливость розгорнути локальний тестовий блокчейн на власній машині, що дуже зручно на перших етапах розробки. Тільки тоді, коли контракти будуть повністю готові та протестовані, їх можна буде розгортати в основній мережі.

Щоб розгорнути локальний блокчейн в данній роботі пропонується програмне забезпечення Ganache.

Ganache — це високоякісний інструмент розробки, зображений на рисунку 2.4, який використовується для запуску власного локального блокчейну для розробки Ethereum. Ganache корисний у всіх частинах процесу розробки. Локальний ланцюжок дозволяє розробляти, розгортати та тестувати свої проекти та смарт-контракти в детермінованому та безпечному середовищі.

Існує дві різні «версії» Ganache, одна настільна програма та одна інструменти командного рядка. Настільний додаток називається Ganache UI і підтримує розробку як для Ethereum, так і для Corda; тим часом інструмент командного рядка називається ganache-CLI, який підтримує виключно розробку Ethereum. Крім того, всі різні версії Ganache доступні для Mac, Windows і Linux.

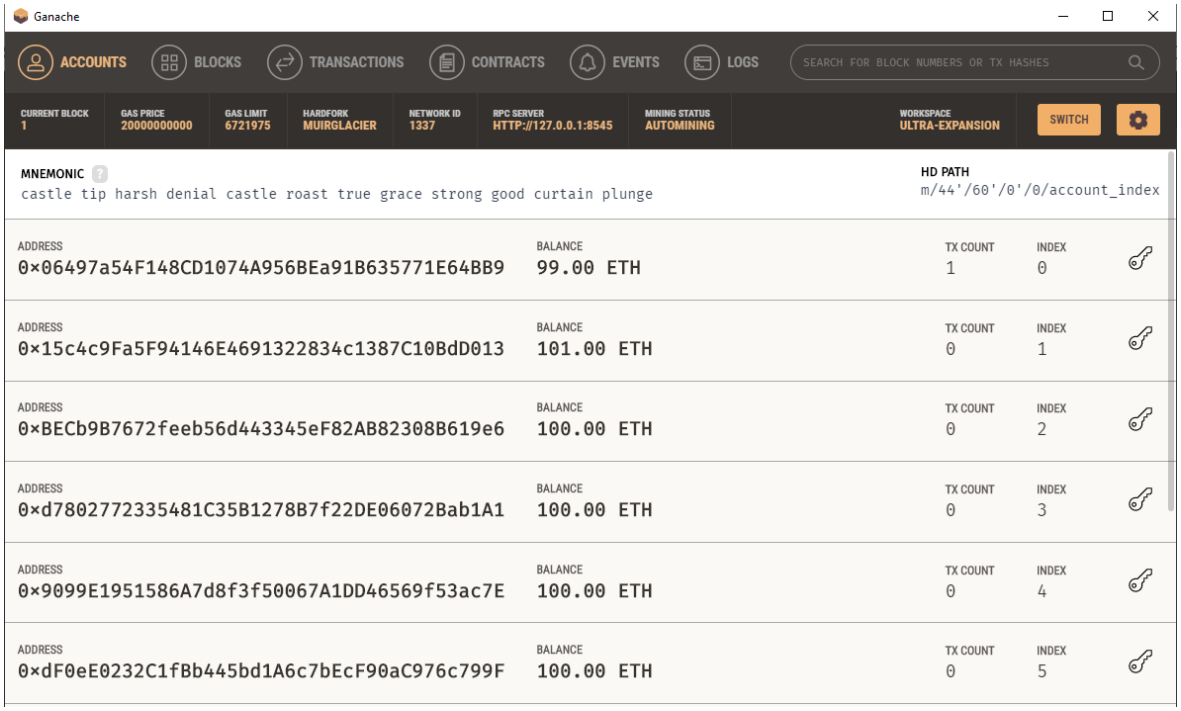

Рисунок 2.4 – Програмне забезпечення Ganache

Є дві основні причини використання Ganache; перший – заощадити гроші, а другий – заощадити час. Під час розробки основними вимогами є рентабельність та ефективність. Тим не менш, завантажуючи смарт-контракти в мережу Ethereum, стягується плата за газ. Це означає, що повинна бути сплачена комісія за транзакцію за кожен смарт-контракт, який розгорнуто. Це може обійтися дуже дорого, оскільки ціни на газ можуть бути досить високими і непередбачуваними. Оскільки це так, набагато вигідніше тестувати смарт-контракти в локальній мережі, уникаючи оплату за транзакції реальними грошима. Це означає, що можна скільки завгодно розгортати контракти, поки вони не стануть бездоганними.

Крім того, потрібен час для завантаження як в основну, так і в тестову мережу Ethereum, і це може бути проблемою, якщо намагатися розробляти контракти швидкими темпами ,чого можна уникнути за допомогою локального блокчейну, налаштованого за допомогою Ganache.

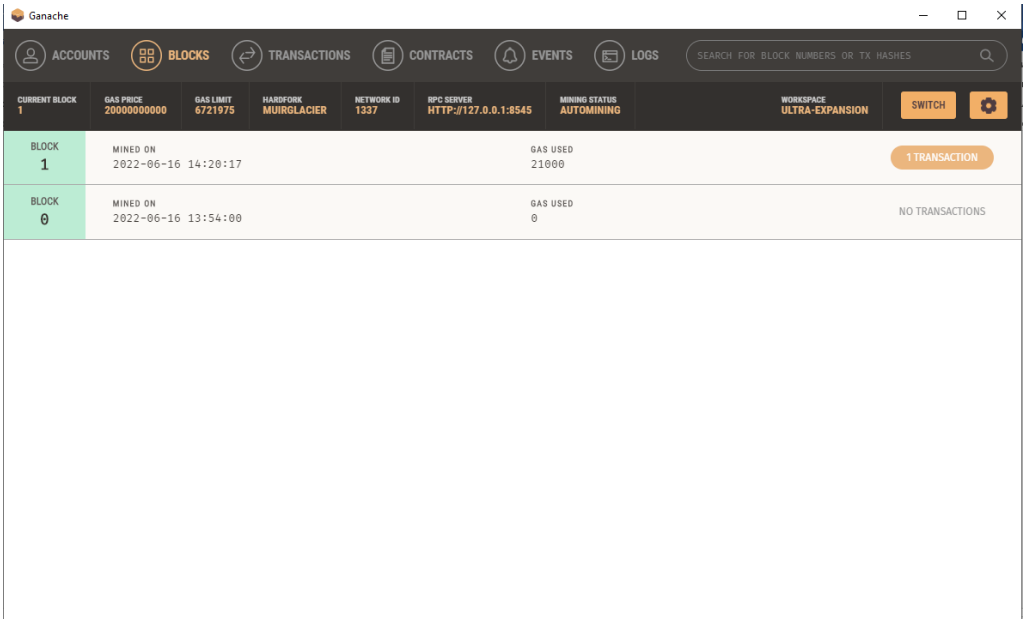

Рисунок 2.5 – Генезис-блок

Після створення робочого середовища потрібно зайти в криптогаманець Metamask. Спочатку потрібно скористатися seed-фразою, яка дається при створені нового блокчейну, та імпортувати її до Metamask, потім зайти в налаштування та додати тестову мережу за допомогою данних з Ganache як показано на рисунку 2.6.

За замовчуванням програмне забезпечення надає доступ до 10 гаманців по 100 ETH на кожному, цього буде достатньо для тестування смарт-контрактів.

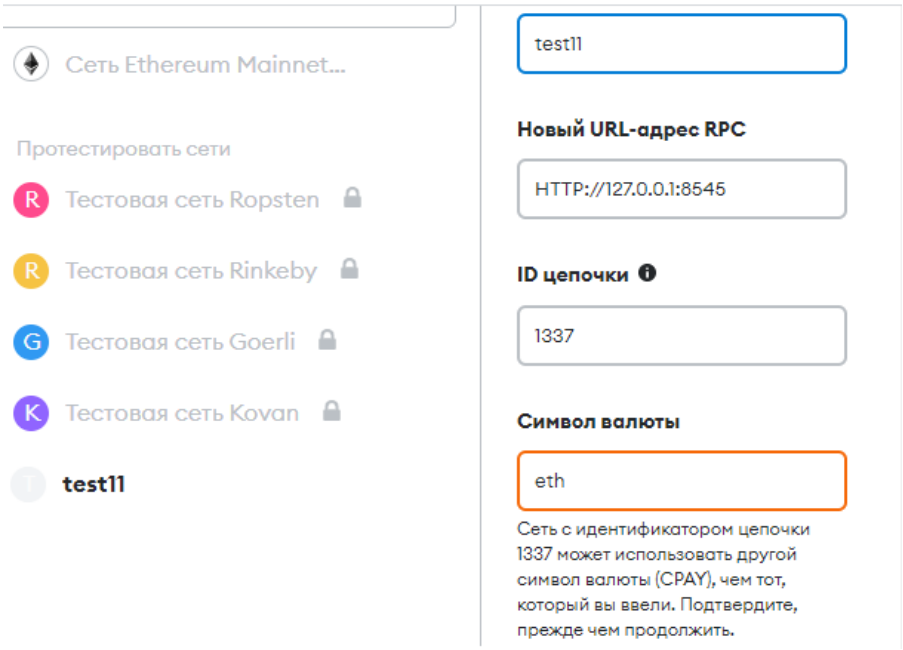

Рисунок 2.6 – Додавання тестової мережі в криптогаманець

На рисунку 2.7 показано підключення Remix IDE до локального блокчейну, після цього можна починати реалізовувати контракти та тестувати їх роботу.

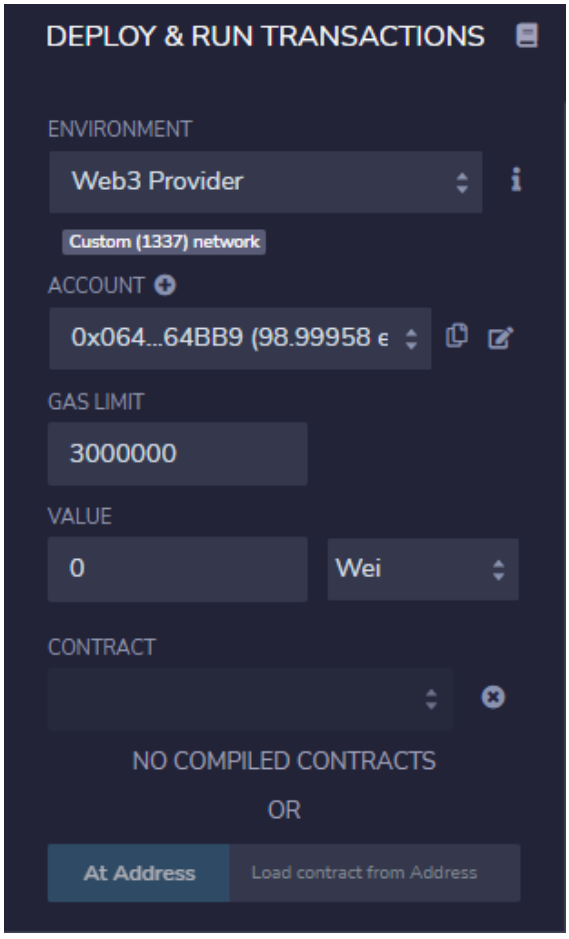

Рисунок 2.7 – Підключення Remix IDE до локального блокчейну

2.4 Проектування смарт-контрактів

2.4.1 Визначення основних смарт-контрактів та правил взаємодії між ними

Згідно концепції, описаної у попередньому розділі, визначимо основні смарт-контракти, які необхідно спроектувати.

Першим розглянемо контракт, призначений для випуску токенів. У зв’язку з появою багатьох неякісних смарт-контрактів для випуску токенів, була зроблена спроба стандартизувати випуск токенів. Зроблено це було в першу чергу, щоб:

– знизити ризики втратити гроші в результаті помилок в смарт-контракті;

–забезпечити можливість стороннім сервісам (біржі, гаманці) безперешкодно взаємодіяти з новими токенами.

Спроектуємо контракт згідно цього стандарту. Для цього необхідно створити файл EIP20Interface.sol, у якому описується інтерфейс контракту, модифікувати файл EIP20.sol під відповідну версію Solidity та безпосередньо створити файл з контрактом для токену.

Варто зауважити, що контракт токена в подальшому може бути модифікований у залежності від необхідних параметрів системи. Крім того, можуть бути створені декілька видів токенів для оплати за різні послуги. Наступними основними контрактами є SocialAssistance (програмує діяльність «Центру надання послуг»), PersonalAssistant (програмує діяльність «Клієнта»), ServiceOrganization (програмує діяльність «Сервісної організації»), PayContractor (контракт-посередник, забезпечує взаємодію клієнта та сервісної організації у момент оплати за послугу).

2.4.2 Опис основних смарт-контрактів

SocialAssistance при своєму створенні розгортає контракт NewToken, при цьому всі токени перераховуються йому на рахунок. У подальшому приймає і обробляє запити від ServiceOrganization та PersonalAssistant.

Сервісна організація реєструє свої послуги, а у відповідь отримує політики, яким повинен відповідати клієнт при запиту на їх отримання. SocialAssistance веде облік переліку сервісних організацій та послуг, які вони надають. Таким чином, клієнт через PersonalAssistant може отримати перелік організацій-надавачів послуг та список послуг, які вони надають.

SocialAssistance на запит від PersonalAssistant реєструє клієнта, визначаючи його категорію і встановлюючи йому політики відповідно до неї. Також передає необхідну кількість токенів клієнту для майбутніх розрахунків за послуги. На запит про повернення токенів від ServiceOrganization перераховує йому кошти за надані послуги. PersonalAssistant реалізує взаємодію клієнта із системою.

PersonalAssistant може взаємодіяти з ServiceOrganization, направляючи йому запит на перелік послуг, які надаються організацією. При запиті на надання послуги передаються політики клієнта, по яких ServiceOrganization оцінює можливість надання йому послуг. Якщо політики клієнта не співпадають з політиками послуги, запит буде відхилено.

У разі підтвердження надання послуги ServiceOrganization, контракт PayContractor забезпечує резервування токена від клієнта на оплату за послугу. Після того, як послуга буде надана і обидві сторони підтвердять це, токен перераховується ServiceOrganization.

Висновки по розділу 2

У даному розділі проаналізовано види токенів та описано концепцію надання соціальних послуг з використанням токенів. Також визначено основні контракти, які потрібно реалізувати для тестування запропонованої концепції, коротко описано алгоритми роботи кожного з них. Для тестування роботи обрано блокчейн Ethereum, для створення контрактів під нього – мову програмування Solidity.

3 РЕАЛІЗАЦІЯ СМАРТ-КОНТРАКТІВ

3.1 Створення смарт-контрактів

У попередньому розділі було описано розгортання середовища для розробки контрактів, запуск локального блокчейн на базі Ethereum та вибрана мова програмування Solidity.

Задля забезпечення функціонування системи, що розробляється, повинні бути розроблені такі смарт-контракти: SocialAssistance, ServiceOrganization, PersonalAssistant, PayContractor, EIP20Interface, EIP20 та NewToken.

Для початку потрібно реалізувати випустк службового токена. Для цього потібно реалізувати декілька контрактів, які відповідають за прийнятий стандарт створення токену. Вони відображені на рисунках 3.1 та 3.2.

```
// Abstract contract for the full ERC 20 Token standard
// https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
pragma solidity ^0.5.0;
interface EIP20Interface {
/* This is a slight change to the ERC20 base standard.
function totalSupply() constant returns (uint256 supply);
is replaced with:
uint256 public totalSupply;
This automatically creates a getter function for the totalSupply.
This is moved to the base contract since public getter functions are not
currently recognised as an implementation of the matching abstract
function by the compiler.
*/
/// @param _owner The address from which the balance will be retrieved
/// @return The balance
function balanceOf(address _owner) external view returns (uint256 balance);
/// @notice send `_value` token to `_to` from `msg.sender`
/// @param _to The address of the recipient
/// @param _value The amount of token to be transferred
/// @return Whether the transfer was successful or not
function transfer(address _to, uint256 _value) external returns (bool success);
/// @notice send `_value` token to `_to` from `_from` on the condition it is approved by `from`
/// @param _from The address of the sender
/// @param _to The address of the recipient
/// @param _value The amount of token to be transferred
/// @return Whether the transfer was successful or not
function transferFrom(address _from, address _to, uint256 _value) external returns (bool success);
/// @notice `msg.sender` approves `_spender` to spend `_value` tokens
/// @param _spender The address of the account able to transfer the tokens
/// @param _value The amount of tokens to be approved for transfer
/// @return Whether the approval was successful or not
function approve(address _spender, uint256 _value) external returns (bool success);
/// @param _owner The address of the account owning tokens
/// @param _spender The address of the account able to transfer the tokens
/// @return Amount of remaining tokens allowed to spent
function allowance(address _owner, address _spender) external view returns (uint256 remaining);
// solhint-disable-next-line no-simple-event-func-name
event Transfer(address indexed _from, address indexed _to, uint256 _value);
event Approval(address indexed _owner, address indexed _spender, uint256 _value);
```

Рисунок 3.1 – Код контракту EIP20Interface.sol

```
pragma solidity ^0.5.0;

import "./EIP20Interface.sol";

contract EIP20 is EIP20Interface {

uint256 constant private MAX_UINT256 = 2**256 - 1;
mapping (address => uint256) public balance;
mapping (address => mapping(address => uint256)) public allowed;
///tated anoute of tokens
uint256 public totalsupply;
address public owner;

///ine following variables are OPTIONAL vanities. One does not nave to include thes.
///They allow one fo custonise the token contract & in no wey influences the core Functionality.
///Sone hallets/ interfaces mignt not even sotner to look at this inforeation.

constructor (
    uint256 _initialAmount,
    string memory _tokenName,
    uint8 _decimaluints,
    string memory _tokenSymbol,
) public {
    balances[msg.sender] = _initialAmount;
    totalsupply = _initialAmount;
    name = _tokenName;
    decimals = _decimaluints;
    symbol = _tokenSymbol;
    owner = msg.sender;

}
function balanceOf(address _owner) public view returns(uint256 balance){
    return balances[_owner]
}
function allowance(address _owner, address _spender) public view returns (uint256 remaining) {
    return allowed[_owner][_sender]
}
```

Рисунок 3.2 – Код контракту EIP20

Далі на основі даних контрактів буде реалізовано випуск власного службового токена для оплати користувачами послуг.

```
pragma solidity ^0.5.0;
import "./EIP20.sol";
contract NewToken is EIP20 {
uint256 constant public weiPerToken=100;

bool private isSelling;

uint256 constant reservPart=1001;
constructor() EIP20(10000,"Test token v.1", 0,"TTn") public {
isSelling=false;
}

function () external payable {}

modifier onceOnly {
require(isSelling == false);

14

isSelling=true;
_;
isSelling=false;
}

function getETHBalance() public view returns(uint256) {
return ((address(this)).balance);
}

function freeTokens() public view returns(uint256) {
return (balanceOf(owner)-reservPart);
}
function calcAmount(uint256 _valueWEI) public pure returns(uint256) {
return (_valueWEI/weiPerToken);
}



}
```

Рисунок 3.3 – Код контракту NewToken

Тепер потрібно реалізувати автоматизовану систему взаємодії між регулюючим органом, організаціями що надають послуги та громадянами. Це реалізується в контрактах: SocialAssistance, ServiceOrganization, PersonalAssistant, PayContractor.

Смарт-контракт SocialAssistance, що програмує діяльність центру надання послуг реєстує організації та встановлює їм певні політики, на рисунку 3.4

```
pragma solidity ^0.5.0;

contract SocialAssistance{

    function servise_register(string memory _name) public {
    organization memory o = organization[msg.sender];
    require(keccak256(abi.encodePacked(_name)) != keccak256(""));
    require(!(d.id > address(0x0)));

    organization[msg.sender] = organization({
        name:_name,
        type_service: _type_service;
        id:msg.sender,

    })


}

}
```

Рисунок 3.4 – Код контракту SocialAssistance

Смарт-контракт ServiceOrganization, що програмує діяльність сервісої організації, реєструє свої послуги в сервісному центрі та отримує певні політики, яким повинен відповідати громадянин, після цього перевіряє користувача на відповідність політикам та приймає рішення чи надавати послуги. Код контракну відображений на рисунку 3.5

```
pragma solidity ^0.5.0;

contract ServiceOrganization{
    function AccessToService(address client_id) public checkClient(msg.sender) checkOrganization(organization_id){
        client storage c = clients[msg.sender];
        organization storage o = organizations[organization_id];
        require(policy[msg.sender][organization_id] < 1);

        uint pos = c.organization_list.push(organization_id);
        gpos = pos
        policy[msg.sender][organization_id] = pos;
        o.client_list.push(msg.sender)


    }
}
```

Рисунок 3.5 – Код контракту ServiceOrganization

Смарт-контракт PersonalAssistant, що програмує діяльність клієнта, реєструє нових користувачів та надає їм певні політики, після чого користувач може вибрати сервісну організацію, до якої йому зручно звернутися, код контракту реалізовано на рисунку 3.6

```
pragma solidity ^0.5.0;

contract PersonalAssistance{
    function client_register(string memory _name, uint _age) public {
        client memory c = clients[msg.sender];
        require(keccak256(abi.encodePacked(_name)) != keccak256(""));
        require((_age > 0) && (_age < 100));
        require(!(d.id > address(0x0)));

        clients[msg.sender] = client({
            name:_name;
            age:_age;
            id:msg.sender;
            organization_list: new address[](0)});
    }
}
```

Рисунок 3.6 – Код контракту PersonalAssistance

3.2 Аналіз запропонованого рішення та подальший розвиток роботи

Проведене дослідження показало, що данна система є працездатною і дозволила вирішити три найбільш значні проблеми, які існують зараз в соціальній сфері, це корупція – прозорість процесу та його автоматизованість якщо не унеможливлюють, то значно перешкоджають зловживанням посадовими особами, значна бюрократична тяганина відтепер в минулому, так як користувач відразу отримує певні політики і може претендувати на певну послугу відразу, в майбутньому ця система може бути модернізована і використовувати цифрові документи для верифікації особи, що збільшить безпеку та точність розподілу соціальних благ, і врешті розроблена система максимально автоматизує весь процес, зменшивши адміністративний персонал, задіяний в данному питанні до мінімуму.

Запропоноване рішення є актуальним і доречним в ситуації, котра скалася в Україні, і подальший розвиток та удосконалення дослідження зможуть створити стійку та злагоджену систему надання соціальної допомоги у нашій державі.

Висновки по розділу 3

У даному розділі запропоновано код реалізації смарт-контрактів для розробленої концепції. Варто зауважити, що представлена у цьому розділі реалізація контрактів є базовою та тестовою і у подальшому функціонал контрактів може бути розширений. Тим не менше, розробка продемонструвала працездатність запропоновної схеми.

ВИСНОВКИ

На початку дослідження було представлено аргументацію актуальності вирішення проблеми надання соціальної допомоги при існуюсій системі. Було виділено три ключових проблеми, які існують в цій свері, а саме корупція, бюрократичне навантаження та непрозорість системи. Провівши роботу над розробкою та створивши власну концепцію надання послуг з використанням сучасних інформаційних технологій, таких як блокчейн, вдалося вирішити всі знайдені слабкі місця.

Виділення коштів на спослуги малозебезпеченому населенню – це справа політиків та адміністрацій, вони вирішують, яку частину бюджету виділяти під ту чи іншу соціальну програму, тим більше які категорії громадян підпадають під її отримання. Це все складні економічні рішення щодо розподілу цих коштів і доступу до них у пересічного громадянина немає.

Проте, провівши дану розробку, можна з певненістю сказати, що вона могла б внести свій вклад в ефективність розподілу уже наявних ресурсів, в зв’язку з тим, що як мінімум ускладнює зловживання посадовими особами своїм службовим становищем, а сам блокчейн як прозора система надає доступ кожному пересвідчитися, куди й кому надходить соціальна допомога. З іншого боку було згадано про тяганину з документами, яка не завжди є простою і зрозумілою для простих людей, так і в звязку з бойовими діями це тільки поглиблило проблему. У данній роботі запропоновано використати певні політики, які встановлюються організації та громадянам, і таким чином здійснювати підтвердження, що саме цей клієнт потребує та має право на допомогу. Однак, в майбутньому данні політики могли б мати не таке абстракне значення, а, наприклад, можуть бути інтегровані в систему верифікацію по цифровим документам, щоб ще більш унеможливити маніпуляції в корисливих цілях.

Також у роботі запропонована система автоматичної оплати за послуги за допомогою службових токенів та зручний й ефективний моніторинг за процесом

через блокчейн, що дозволяє вивільнити значний адміністративний та організаційний ресурс, більшість процесів в данному ланцюжку може бути автоматизована. Дане дослідження має багато перспектив та цікавих напрямків для розвитку, адже застосування такої технології в розподілі соціальної допомоги не єдинний спосіб її застосування.

ПЕРЕЛІК ПОСИЛАНЬ

1. Блокчейн – огляд, характеристика. URL: https://uk.wikipedia.org/wiki/Блокчейн (веб-ресурс)

2. Криптовалюта – загальні відомості, використання та значення. URL: https://uk.wikipedia.org/wiki/криптовалюта (веб-ресурс)

3. Фіатні гроші – загальна характеристика. URL: https://uk.wikipedia.org/wiki/Фіатні_гроші (веб-ресурс)

4. Інтернет речей – сучастий стан,проблеми. URL: https://uk.wikipedia.org/wiki/Інтернет_речей (веб-ресурс)

5. Застосування технології в Україні. URL: https://uk.wikipedia.org/wiki/CETAM#Blockchain (веб-ресурс)

6. *Melanie Swan*. Blockchain: Blueprint for a New Economy. — 2015. — 152 p. — ISBN 978-1-4919-2047-3. (книга, один автор)

7. Nakamoto, Satoshi (2008). Bitcoin: A Peer-to-Peer Electronic Cash System (Стаття, один автор)

8. Як створюється блок і хто отримує нагороду. URL: https://bitstat.top/blog.php?id_n=2196 (веб-ресурс)

9. Block Height. Guide to Blockchain. URL: https://www.investopedia.com/terms/b/block-height.asp (веб-ресурс)

10. Ethereum – Глобальна мережа з відкритим кодом для децентралізованих застосунків. URL: https://ethereum.org/ru (веб-ресурс)

11. Що таке смарт-контракти? URL: https://habr.com/ru/post/448056 (веб-ресурс)

12. Solidity. URL: https://docs.soliditylang.org/en/latest/ (веб-ресурс)

13. RemixIDE. URL: https://remix. ethereum.org

14. White Paper GitHub URL: https://github.com/ethereum/wiki/wiki/White-Paper (веб-ресурс)

15. Smart Contracts in 2021: What it is & Why matters? URL: https://research.aimultiple.com/smartcontracts/ (веб-ресурс)

16. Романьков, В.А. Введення в криптографію. Курс лекцій / В.А. Романьков. — М.: ФОРУМ, 2012. — 240 с (Лекційні матеріали)

17. ОСНОВИ ШИФРУВАНЯ (ЧАСТИНА 1) - АЛГОРИТМ ДИФФИХЕЛЛМАНА Securitylab - 25.01.2016 URL: https://www.securitylab.ru/analytics/478912.php (веб-ресурс)

18. Транзакції Ethereum: як працюють і як провірити. URL:

https://coinpost.ru/p/tranzakcii-ethereum-kakrabotayut-i-kak-proverit (веб-ресурс)