НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

«На правах рукопису»
УДК 004.942:519.216.3

«До захисту допущено»
Завідувач кафедри
_____ О. Л. Тимощук
«___» _____2018 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 124 Системний аналіз

на тему: «Система керування коштами та дольовою участю співвласників підприємства на базі розподіленого реєстру»

Виконав:
студент ІІ курсу, групи КА-72мп
Ярославський Володимир Олександрович _____

Керівник:
к.т.н., доцент Дідковська Марина Віталіївна
_____

Рецензент:
к.т.н., доцент Заболотня Тетяна Миколаївна _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань
Студент _____

Київ
2018

РЕФЕРАТ

Магістерська дисертація: 110 стор., 39 рис., 21 табл., 1 додаток, 30 джерел.

Об'єктом розробки є смарт-контракт в розподіленій базі даних на основі технології Blockchain.

Метою даного проекту є розробка смарт-контрактів у розподіленій базі даних для веб-додатку, що відповідатиме за розподіл коштів у підприємстві. Вивчення та дослідження технології Blockchain.

В процесі розробки було:

- досліджено технологію Blockchain, структуру організації її блоків та принцип роботи системи;
- розглянуто алгоритми смарт-контрактів і проведено аналіз їх переваг та недоліків;
- спроектовано структуру смарт-контрактів;
- розроблено веб-інтерфейс для керуванням створення смарт-контрактів;
- розроблено модуль взаємодії смарт-контрактів з Веб-додатком.

BLOCKCHAIN, СМАРТ-КОНТРАКТ, ETHEREUM, SOLIDITY, JAVASCRIPT, MONGODB, WEB3, ВЕБ-ДОДАТОК, POW, POS, SOLIDITY, ERC20.

ABSTRACT

The master’s dissertation: 110 p., 39 fig., 21 tabl., 1 appendix, 30 sources.

The object of development is smart contract in a distributed database based on Blockchain technology.

The purpose of this project is the development of smart contracts in a distributed database for the Web application, which will be responsible for distributing funds in the enterprise. Study and research on Blockchain technology.

In the development process was:

- investigated Blockchain technology, the structure of organization of its blocks and the operating principle of the system;
- analyzed algorithms of smart contracts, considered their advantages and disadvantages;
- designed smart contracts structure;
- developed web-interface for smart-contract creation;
- developed a module for an interaction of smart contracts with the Web application.

BLOCKCHAIN, SMART CONTRACTS, ETHEREUM, SOLIDITY, JAVASCRIPT, MONGODB, WEB3 WEB APPLICATION, POW, POS, SOLIDITY, ERC20.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Рівень вищої освіти – другий (магістерський)

Спеціальність (спеціалізація) – Системний аналіз фінансових ринків

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Тимощук О. Л.

«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Ярославському Володимиру Олександровичу

1. Тема дисертації «Система керування коштами та дольовою участю співвласників підприємства на базі розподіленого реєстру», науковий керівник дисертації Дідковська Марина Віталіївна, доцент, к.т.н., затверджені наказом по університету від 07 листопада 2018 р. № 4121-с

2. Термін подання студентом дисертації

3. Об’єкт дослідження: смарт-контракт в розподіленій базі даних на основі технології Blockchain.

4. Предмет дослідження: імплементація системи керування коштами та дольовою участю співвласників підприємства на базі розподіленого реєстру.

5. Перелік завдань, які потрібно розробити: (1) Дослідити особливості технології Blockchain та платформи Ethereum (2) Розробити основні принципи роботи смарт-контракту та веб-інтефейсу до нього (3) Розробити програмний продукт

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: 39 рис., 21 табл.

7. Дата видачі завдання

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Написання вступу магістерської дисертації	07.09.2018—13.09.2018	
2	Вибір технологій написання смарт-контракту та інтерфейсу.	14.09.2018—23.09.2018	
3	Детальний огляд обраних платформ, вивчення документації.	24.09.2018—30.09.2018	
4	Вивчення методів створення приватних блокчейнів	01.10.2018—08.10.2018	
5	Написання програмного продукту та його застосування на приватному блокчейні	09.10.2018—17.10.2018	
6	Підготовка матеріалів першого розділу магістерської дисертації	18.10.2018—26.10.2018	
7	Підготовка матеріалів другого розділу магістерської дисертації	27.10.2018—04.11.2018	
8	Підготовка матеріалів третього розділу магістерської дисертації	05.11.2018—14.11.2018	
9	Підготовка матеріалів розділу стартап-проекту магістерської дисертації	15.11.2018—22.11.2018	
10	Написання висновку магістерської дисертації	23.11.2018—26.11.2018	

Студент Ярославський Володимир Олександрович

Науковий керівник дисертації Дідковська Марина Віталіївна

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Blockchain	– розподілений реєстр, що складається з ланцюжку блоків.
ПЗ	– програмне забезпечення.
Смарт-контракт	– комп'ютерний алгоритм, який виконується на розподілених реєстрах.
EVM	– Ethereum Virtual Machine
ICO	– Initial coin offering
PoS	– Proof of Stake
PoW	– Proof of Work
P2P	– Peer to peer
Solidity	– мова програмування для платформи Ethereum

ВСТУП

Внаслідок технологічного буму в останні роки величезна кількість проблем, що не мали рішення в минулому були або будуть розв’язані найближчим часом.

Наразі до вже існуючих цінностей додаються такі як простота, прозорість та відкритість. І тому не дивно, що розвиток такої технології як Blockchain припав саме на цей час.

Пояснення фундаментальних принципів роботи цього концепту лежить у його назві. “Ланцюг блоків” - це і є Blockchain. Ідея полягає в об’єднанні певного набору інформації (“блоки”) у глобальний репліковний розподілений реєстр (“ланцюг”) [1]. Така проста зовні технологія уже зробила переворот у багатьох сферах людського життя починаючи від фінансів і аж до системи охорони здоров’я.

Через 5 років після старту Bitcoin, криптовалюти, що популяризувала Blockchain, з’явилась технологія, що стала причиною такого стрімкого розвитку технології поза звичного тоді фінансового сектору, зламавши стериотип про те, що Blockchain - це складно і тільки для експертів в банківській сфері і IT. Ця технологія - смарт-контракти. По суті - це алгоритм перерозподілу коштів, який є повністю децентралізований і результат його роботи є загальнодоступний і незмінний. Розумні контракти - це продукт компанії Ethereum, і їх зберігання та виконання забезпечені однойменним блокчейном. Також, разом зі смарт контрактами, дана компанія розробила зручний механізм їх написання, який став спеціалізованою мовою програмування під назвою Solidity.

Весь алгоритм роботи розумного контракту записується в код програми. Сам код зберігається та виконується в мережі Ethereum, дозволяючи

розподіляти транзакції між сторонами без застосування зовнішніх механізмів. Всі транзакції, що проходять через смарт-контракт є незворотними та простежуваними.

Найбазовішою властивістю Blockchain насамперед є те, що інформація, що в нього потрапила, залишиться там назавжди і немає жодних методів обійти це. Тому найперше призначення блокчейну було саме як нове покоління електронної валюти. Іншим важливим застосуванням блокчейну була боротьба проти шахрайства і корупції [2].

Сама криптовалюта (монети) є потужним механізмом для проведення фінансових операцій, так як для неї немає кордонів, вона умовно анонімна та не має фізичного носія, що дозволяє суспільству повністю відмовитись від послуг банків тим самим значно урізавши транзакційні втрати.

У даній магістерській дисертації вивчається застосування Blockchain для підтримання розрахункових операцій всередині підприємства, а також його взаємодія зі сторонніми особами, що, на мою думку є дуже вдалим і доцільним використанням описаної технології.

Одними з основних причин занепаду компаній на ранній стадії є нераціональність, непрозорість керування активами, складність керування та конфлікти між засновниками. Згідно даних за 2016 рік, у світі було створено більш ніж 25000 стартапів, що приймали криптовалюти в якості оплати і 92% з яких наразі неактивні [3]. Дана проблема і є стимулом розвивати системи, подібні до тієї, що описана у даній роботі.

Також, через новизну технології, ступінь її прийняття суспільством поки знаходиться на рівні нижче середнього, тому у випадку створення смарт-контракту виникає потреба в створенні роз’яснень або зручного інтефейсу, який допоміг би розширити коло потенційних користувачів людьми, що мають труднощі у взаємодії зі смарт-контрактами та блокчейном взагалі.

Таким чином, метою роботи є розробка архітектури, опис основних алгоритмів, методів роботи та реалізація розумних контрактів, що включають умови для забезпечення автоматичного розподілу коштів, всі транзакції яких будуть записані в Blockchain.

Для досягнення цих цілей вирішені наступні завдання:

а) Проведено аналіз існуючих підходів для керування коштами і дольовою участю всередині компанії.

б) Розроблено архітектуру смарт-контракту, який знаходитимется на системі розподілених реєстрів і візьме на себе основну функцію з розподілу коштів та акцій.

в) Реалізовано таку систему на платформі Etherium та розроблено зручний веб-інтерфейс для керування створенням контракту.

г) Впроваджено розроблену систему на приватному блокчейні за допомогою методів geth.

Об’єктом дослідження є система керування коштами і дольовою участю всередині компанії на основі розподіленого реєстру.

Предметом дослідження є методи та алгоритми формування системи керування коштами і дольовою участю всередині компанії на основі розподіленого реєстру.

Наукова новизна отриманих результатів полягає в наступному.

- обгрунтовано використання блокчейну в даній задачі для забезпечення повної прозорості дольової учаті та транзакцій;
- розроблено смарт-контракт, що виконує функції, сформульовані даною задачею;
- Описано алгоритм створення приватного блокчейну засобами з відкритим кодом та ліцензіями, що дозволяють вільне комерційне використання, що суттєво знижує вартість впровадження приватного розподіленого реєстру.

Практичними результатами роботи є розробка системи керування коштами і дольовою участю всередині компанії на основі розподіленого реєстру а також веб-додатку, що ініціює створення смарт-контракту, що є основою цієї системи.

Створення веб-інтерфейсу обумовлено тим, що важливо знайти той спосіб, за допомогою якого користування смарт-контрактів буде простим та зручним для середньостатистичного користувача. На основі цього було вирішено створити засіб взаємодії зі смарт-контрактом окрім уже існуючих систем.

1 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ МАГІСТЕРСЬКОЇ ДИСЕРТАЦІЇ

1.1 Аналіз актуальності задачі

На сьогоднішній день невідкладною проблемою є можливість управління та вдосконалення корпоративних фондів, оскільки це найважливіший аспект управління підприємством, і від цього залежить оперативна ефективність компанії.

В епоху технологічної революції можна більш швидко та ефективно здійснювати управління коштами розробляючи нові програмні рішення, які дозволяють зосередити увагу на особливостях корпоративної діяльності з використанням нових технологій.

Управління коштами компанії - безперервний аналіз та контроль грошових потоків для збільшення прибутку компанії, від етапу планування до фактичних транзакцій. Добре відрегульована система забезпечить стабільну платоспроможність організації, можливість управляти вільними коштами та більш точно прогнозувати майбутні витрати та прибутки. За відсутності систем фінансового планування компанії часто стикаються з такими труднощами:

- заборгованість по заробітній платі працівників;
- нестача коштів, призначених для зменшення дебіторської заборгованості, що призведе до накопичення боргів;
- брак оборотного капіталу;
- труднощі в кредитуванні.

Існує багато дискусій щодо сфер використання децентралізованих технологій, окрім фінансів. Зрештою, експерти погоджуються, що можливості та перспективи розвитку технологій криптовалют ще не повністю розкриті.

Одним з найперспективніших напрямків у галузі є впровадження криптографії для бізнесу. Це може бути використано для оптимізації бухгалтерського обліку та відстеження продуктивності праці, або збільшення мотивації працівників в компанії. У такому разі можна довго сперечатися про здатність децентралізованих підприємств працювати в реальних умовах, розкриваючи обидві точки зору. Проте ясно, що без технології, на якій будується криптовалюта, підприємства будь-якого розміру у майбутньому не будуть конкурентоспроможними.

1.2 Аналіз існуючих методів розподілу коштів

Є два основні способи вирішення завдання управління фондами компанії: традиційні та з використанням криптографічних технологій.

Перший - це традиційний метод. До недавнього часу він був майже єдиним. Часто інформація про всі грошові потоки розміщується на спеціальній платформі для цілей бухгалтерського обліку. У країнах СНД найпопулярнішою платформою є 1С. Керівництво має призначити людину з професійною освітою звітувати про всі фінансові потоки, але тут виникає людський фактор. Крім того, всі транзакції мають бути записані. З цих причин на сьогодні електронний облік документів не є популярним, тому поки більшість компаній організовують документообіг в паперовому вигляді. Це може призвести до помилок і неточностей, а найважливішим питанням є реалізація умов договору та управління. Існують також обов'язкові обмеження, які прив’язують підприємство до банків, банківських рахунків, комісійних та затримок на транзакцію.

Ця система застаріла і неефективна для нашого часу. Для цього потрібні новаторські втручання та нові шляхи вирішення існуючих проблем.

Другий спосіб полягає в застосуванні криптографії. Це дозволяє спростити бухгалтерський та фінансовий облік, а основне - підтримувати максимальну прозорість діяльності підприємства. Якби існували ідеальні умови для ведення бізнесу, які були б на основі системи Blockchain, тоді складні принципи обліку просто б зникнули.

Особлива увага також приділяється стосункам акціонерів або відносинам між роботодавцями та працівниками. Ці технології дозволяють організувати та автоматизувати процес перерозподілу дольової участі за допомогою розумної контрактної системи. У випадку відносин між власниками і найманими працівниками, в якому другим створюється уявлення, що вони є частиною компанії тому їх прибуток зростатиме зі зростанням компанії, зроблять ці уявлення більш реалістичними.

Для абсолютної прозорості операційного доходу підприємтву вистачає лише кілька криптовалютних гаманців, які отримують всі надходження. Таким чином, кожен співробітник може взнати фінансове становище підприємства та його перспективи на ринку.

У системі виплат заробітної плати на підприємствах можна використовувати приватну валюту, яку в будь-який момент можна обміняти на державну (фіатну) валюту. Працівники також можуть стати міноритарними акціонерами компанії, в якій вони працюють. Наприклад, залишаючи частину доходу на вашому криптовалютному рахунку, ви дозволяєте компанії отримати вільні гроші в обігу за певні дивіденди.

1.3 Криптовалютні технології в промисловості

Сьогодні більшість великих компаній зі списку Fortune 500, від фінансів, роздрібної торгівлі до автомобільних та космічних компаній вивчають технологію Blockchain, яка забезпечує безпеку в комерційних угодах і захист від конкуруючих підприємств. У 2017 р. відбулась хвиля досліджень та розвитку Blockchain з кількома відомими доказами концепції (PoC) та пілотними програмами в різних галузях. PoC - це потенціал для реалізації певного методу або ідеї, щоб довести його здійсненність або продемонструвати конкретну концепцію, в теорії і на практиці [4].

Темп наукових досліджень і розробок пришвидшився, і вони розширюють поле діяльності на інші сфери підприємництва. Хоча фінансовий сектор завжди був провідним дослідником Blockchain технологій, існує цілий ряд галузей в сфері роздрібної торгівлі, судноплавства, телекомунікацій, аерокосмічній, автомобільній та інших, які наздоганяють фінансистів за кількістю розробок і прототипів.

Британський венчурний фонд Outlier Ventures випустив дайджест, який описує діяльність, пов’язану з технологією Blockchain, близько 285 компаній. У ньому присутні близько 140 компаній, що проводять PoC, від ABN Amro до Wells Fargo. Тут також перелічено 15 компаній, які подали заявку на патенти, включаючи Amazon, Boeing, IBM та Western Union.

Крім того, деякі компанії проявляли інтерес до технології шляхом інвестицій та поглинань інших компаній. В 2016 - 2017 роках Airbnb, Daimler, та інші придбали Blockchain-стартапи.

Важливо відзначити, що компанії ще не передають всі свої бізнес-процеси на Blockchain. Як і в багатьох проектах з розробки Blockchain-додатків (DApps), потребується багато дослідницької роботи та інвестицій, але тільки

деякі з них є комерційно життєздатними та готові прийняти виробничі процеси.

Фінансові послуги є найближчими до досягнення Blockchain-бізнес-процесів, прикладами яких є: Австралійська фондова біржа та Pioneer Group, яка оголосила про успішний пілотний проект та дослідження після запуску Blockchain в 2018 році [5].

Якщо у всьому світі є зацікавлені компанії, технологію Blockchain можна використовувати для створення нових можливостей та інструментів. Blockchain може допомогти оптимізувати витрати, покращити певні процеси, відстежувати продуктивність роботи та клієнтів, покращити безпеку продукту та зменшити шанс підробок.

З розвитком бездротових сенсорних мереж, технології штучного інтелекту та промислової інфраструктури Blockchain може потенційно заохотити підприємства до типу ведення бізнесу "Industry 4.0".

1.3.1 Недоліки Blockchain при його впровадженні на підприємства

Розглядаючи можливість використання технології Blockchain, компанія стикається з багатьма проблемами та труднощами. По-перше, багато основних технологій Blockchain ще не готові, або вони не були випробувані на широкомасштабній комерційній реалізації. Окрім безпеки транзакцій та масштабованості, компанія стикається з проблемами при напрацюванні бази знань про розподілені реєстри та наймі досвідчених розробників.

Припускаючи, що технічні проблеми вирішені, компанії повинні зрозуміти ключові компроміси, а саме - в централізованих системах вище

конфіденційність данних та час обробки транзакції, в той же час як розподілені реєстри несуть в собі повну безпеку данних та їх цілісність.

Тоді компанія повинна репозиціонувати себе для роботи з конкурентами, постачальниками та замовниками в ланцюжку набуття доданої вартості. У деяких випадках їй може знадобитися внести цінні пропозиції та переконати всі сторони працювати разом, щоб скористатися перевагами нових технологій.

За допомогою технології Blockchain створюються надії і очікування поки нереальних технічних рішень, а також нових можливостей.

Імовірно спільнота Blockchain не зможе вирішити ці проблеми, і ця технологія не зможе довести високі очікування покладені на неї.

1.4 Існуючі рішення

Концепція розумного контракту з’явилась тільки в 2013 році і в даний час проходить етап розповсюдження. Все нові і нові проекти створюються за допомогою цієї технології щодня (Рис. 1.1). З огляду на переваги смарт-контрактів, вони можуть стати найціннішими додатками технології Blockchain у довгостроковій перспективі. З часом вони почнуть серйозно конкурувати з традиційними контрактами.

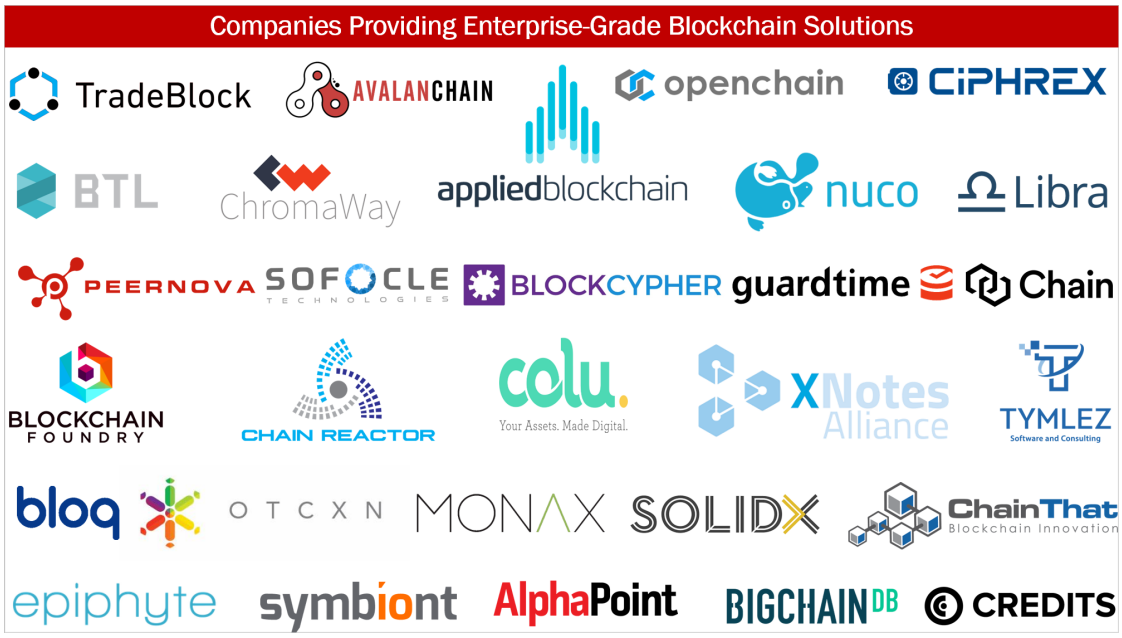

Рисунок 1.1 – Blockchain-компанії

Задачі, які вирішуються в цій роботі - дуже складні у реалізації на підприємстві. Тому необхідно провести детальне вивчення фінансової частини компані1, провести поглиблений аналіз і забезпечити продукт, найбільш життєздатний в реальності. Для цього рішення поки що немає. Але, враховуючи нагальність проблеми, є кілька подібних проектів, які є потенційними рішеннями.

Перший - Bitwage. Продукт цієї компанії дозволяє легко та безпечно розподіляти заробітну плату більшості працівників компанії. Міжнародна система заробітної плати використовує криптовалюти для передачі коштів з одного місця в інше, щоб потім конвертувати її в державну валюту. Тому Bitwage відіграє нематеріальну посередницьку роль у передачі заробітної плати. Ця компанія була створена в 2014 році і наразі Bitwage має 18 000 активних користувачів, з оборотом в понад 30 млн доларів США. В даний час платформа підтримує 25 різних валют [6].

Для зручності міжнародних великих компаній, які переходять на розрахунки в криптовалюті, оплата здійснюється протягом декількох хвилин, а

не протягом 2-4 робочих днів. Крім того, надсилання міжнародних фінансових транзакцій таким способом є дешевшим. Вам більше не потрібен банківський рахунок отримання або відправлення коштів.

У випадку переведення в традиційний спосіб з банку однієї країни в іншу, обидва банки візьмуть певну комісію за свої послуги, а працівник заплатить за конвертацію з однієї валюти в іншу. Взагалі, переваги Bitwage є очевидними у даній ситуації.

Другою послугою є BitShares. Це відкрита фінансова платформа, що працює у режимі реального часу з відкритим кодом на базі Blockchain. Він забезпечує децентралізований обмін криптовалют без потреби в централізованому активі цільових фондів, як, наприклад, на Нью-Йоркській фондовій біржі. Власна криптовалюта “BTS", яка є внутрішньою розробкою компанії, використовується для оплати мережевої активності або сторонньої підтримки BitShares [7].

Третім сервісом є Otonomos. Це інструмент для використання Blockchain для власників компаній на основі їхніх акцій, щоб, використовуючи смарт-контракти, керувати їхніми активами в компаніях. Створена в кількох основних юрисдикціях світу для фінансування та управління компаніями [8].

Четверта компанія - AIRA. Її продукт дозволяє управляти децентралізованими саморегуляторними організаціями, її агентами, контрактами та створювати ассети, іншими словами - автоматизувати бізнес-процеси. Мережа Etherium, яку використовує компанія дозволяє створювати монети (одиниця криптовалюти) та токени (цифрові цінні папери, що показують володіння деякою цінністю).

Творці AIRA віддають важливу роль майбутньому штучного інтелекту в роботі децентралізованого підприємста. Згідно з проектом, штучний інтелект та машини / комп'ютери, об’єднані у децентралізовану мережу зменшать

витрати на технічне обслуговування та створять інструменти запобігання ризикам, пов'язаним з використанням звичайних (централізованих) систем [9].

Висновки за розділом 1

Метою першого розділу магістерської роботи є вивчення проблем, пов'язаних із фінансовою діяльністю компаній, ризики, які вони можуть спричинити, технологій розподілення коштів та акцій всередині підприємств а також існуючих рішень, які вирішують ці проблеми.

Крім того, метою даного дослідження є вивчення теоретичних основ вказаних систем та дослідження впливу різних факторів.

Загальна мета цієї магістерської дисертації полягає в розробці системи автоматизації фінансових операцій підприємства та створення відповідних розумних контрактів на Blockchain. Система повинна бути повністю готовою до впровадження в реальну роботу підприємства, а також відповідати стандартам продуктивності та стабільності роботи.

2 ТЕХНОЛОГІЯ BLOCKCHAIN. ЗАСТОСУВАННЯ ПЛАТФОРМИ ETHEREUM ДЛЯ СМАРТ-КОНТРАКТІВ

2.1. Blockchain. Плюси та мінуси

Блокчейн - це тип зберігання даних, який дозволяє цифровим чином ідентифікувати та відслідкувати транзакції. Передаючи цю інформацію по розподіленій комп'ютерній мережі, створювати розподілену мережу довіри в якомусь сенсі (рис. 2.1) [10].

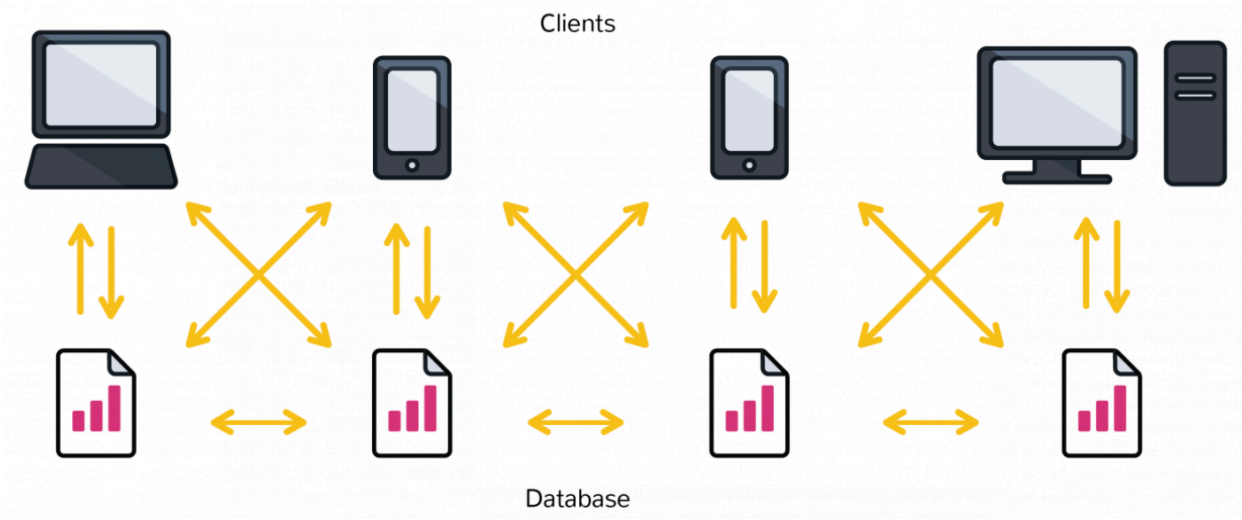

Рисунок 2.1 – Взаємодія мобільних та комп’ютерних приладів з розподіленою БД

З прикладної точки зору блокчейн - це інновація, що базується на трьох концепціях: мережі P2P, асиметричній криптографії та розподіленому консенсусі на основі вирішення математичних задач. Ці ідеї не є новими.

Блокчейн може розглядатися як синхронізована БД з такою кількістю копій, як і вузлів в мережі; як суперкомп'ютер, сформований складовими з усіх центральних та графічних процесорів, що належать до нього. Він використовується для збереження і обробки інформації, або як API. Різниця в тому, що вам не потрібно розробляти бекенд, і ви можете бути впевненими, що

всі дані добре захищені та правильно оброблені в мережі. Можна припустити, що Blockchain - це журнал, в якому факти об'єднуються в ланцюг однотипних вузлів.

Мережі P2P та інші розподілені системи повинні розв’язувати непросту проблему: вирішення конфліктів або їх аналіз та узгодження. Реляційні БД забезпечують цілісність через посилання, але така функціональність не існує в розподілених системах. За умови, що 2 несумісні факти надходять одночасно, система повинна мати інструкцію, щоб визначити, який факт є правильним.

Основною ідеєю блокчейн є надання єдиної точки доступу, до великої кількості даних для всіх учасників. Сатоші Накамото запропонував в 2008 році таку концепцію. У 2009 році була перша реалізація на практиці - Біткойн - цифрова валюта [11].

Дані в блокчейн збираються та поступово накопичуються, створюючи певний журнал з даними, що постійно доповнюється (мал. 2.2). В даний журнал не можна вносити які-небудь зміни або видаляти хоч якусь інформацію. Туди можна записати скільки завгодно транзакцій, що робить таку технологію унікальною.

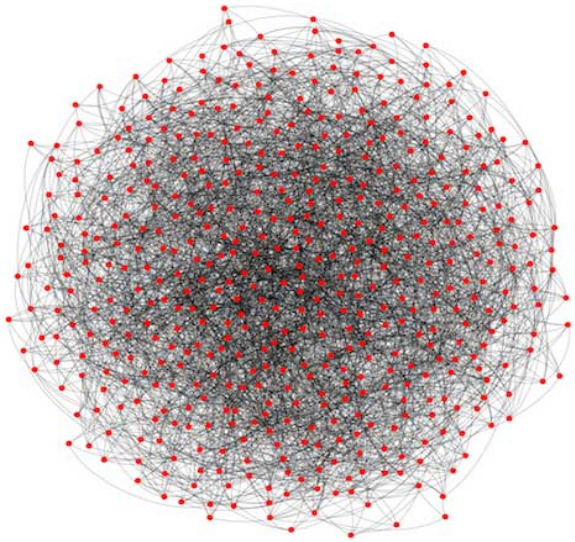

Рисунок 2.2 – Візуалізація децентралізованого Блокчейн

Щоб затвердити транзакцію, її формат та підписи повинно бути перевірено, а потім записано в блок. Факти згруповуються в блоки, і в мережі залишається лише один блоковий ланцюжок. Кожен новий блок має посилання до попереднього блоку (мал. 2.3). Перш ніж додавати до блоку, факти розглядаються як неузгоджені [12].

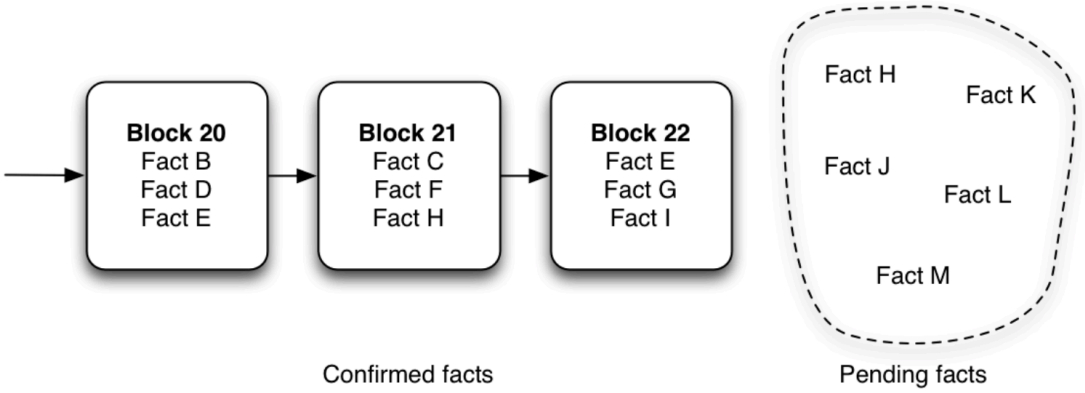

Рисунок 2.3 – Блоки в технології блокчейн

Блок будується з назви і списку транзакцій. Назва складається з свого хеш-значення, хеша попереднього блоку, хешів транзакцій і додаткової інформації (мал. 2.4). Перша транзакція - отримання комісії, що потім стане винагородою для юзера, який створив блок [12].

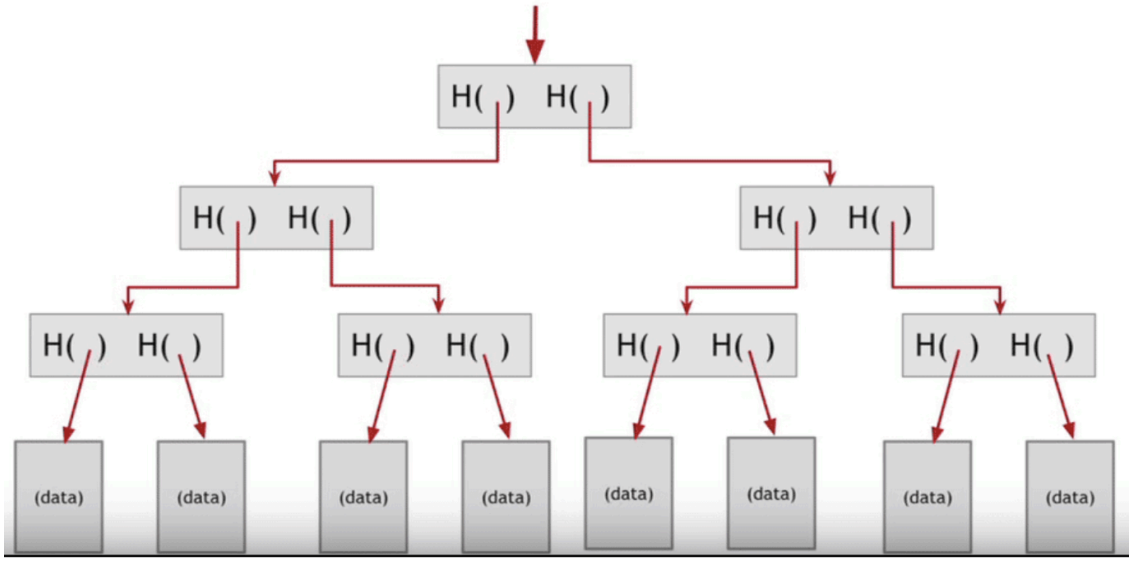

Рисунок 2.4. – Дерево Меркеля

2.1.1. Хеш-функція

Хешування - алгоритм, що бере на вхід інформацію різної довжини та розміру і повертає строку певної довжини, визначеної даним алгоритмом. У контексті цифрових валют, наприклад біткойн, транзакція після хешування виглядає як послідовність символів конкретної довжини (рис. 2.5) [13].

INPUT	HASH
Hi	639EFCD08ABB273B1619E82E78C29A7DF02C1051B1820E99FC395DCAA3326B8
Welcome	53A53FC9E2A03F9B6E66D84BA701574CD9CF5F01FB498C41731881BCDC68A7C8

Рисунок 2.5 – Результат хешування

Крипто хеш-функція - це особливий тип хеш-функції, що має різні атрибути, потрібні криптографії. Є ряд властивостей, що необхідні аби вважатися безпечною хеш-функцією[13]:

- детермінування – це означає, що завжди буде одержано один результат, незважаючи на те, скільки разів проводився аналіз певного входу через хеш-функцію;
- швидкий розрахунок – хеш-функція повинна швидко повертати хеш-вхід. Якщо процес повільний, система стане неефективною;
- складність оберненого обчислення - означає, що беручи H(A) нереально визначити A, H(A) – хеш, A – дані, що вводяться;
- тривіальні зміни в даних, що вводяться призводять до зміни хешу. Тому будь-які незначні зміни у дані ставатимуть значними;

- колізійна стійкість. Через 2 різних типи вихідних даних A і B - H(A) не може дорівнювати H(B), тобто кожен вхід має свій інший хеш, де H(A) і H(B) - хеші від A і B;
- загадка. Для будь-якого виходу “Y”, якщо k вибирається з розподілу з високою мін. ентропією, неможливо знайти вхідні дані x такі, що H (k | x) = Y.

2.2. Публічний та приватний ланцюжки блокчейн

Існує два види “ланцюга” [14]:

- Public Blockchain - це БД з відкритим кодом. Прикладом цього типу є біткоїн. Кожен учасник має можливість записати та прочитати дані;
- Private Blockchain – вже з’являються обмеження на запис або читання. Крім того, можуть бути встановлені пріоритетні вузли. В приватному блокчейні заздалегідь створюється група людей та виконує обробку транзакцій.

2.2.1. Подібність ланцюгів

Впродовж років розвивалися різні типи блоків, і термінологія часто виявлялась неправильно. Це трапляється через те, що в загальнодоступних та приватних мережах безліч спільного:

- вони децентралізовані однорангові, всі учасники обох ланцюгів підтримують репліку загального реєстру;

- за допомогою протоколу, що називається консенсусом підтримують репліки синхронно;
- Якщо хоч якісь учасники є помилковими або зловмисними, ланцюги забезпечують гарантію неперервності та незмінності ланцюга.

2.3. Використання блокчейну іншими криптовалютами

Незважаючи на те, що криптовалюти використовують технологію блокчейн, вони використовують різні підходи. Через те, що першою криптовалютою був біткоїн, він був в певній мірі першопрохідцем, і після аналізу його роботи були розроблені інші, так звані, альткоїни, аби покращити роботу біткоїн.

У випадку Bitcoin новий блок додається з інтервалом раз в десять хвилин. Він перевіряє та записує транзакції. Для цього майнери користуються надзвичайно потужним обладнанням, щоб забезпечити розрахунок, який знаходить потрібне число. Так як кількість транзакцій біткойну постійно росте, але залишається обмеження в десять хвилин для створення блоку, це може нести за собою більше часу для підтвердження, що спричиняє сповільнення роботи системи.

До прикладу, в Litecoin це займає 2,5 хвилини, щоб створити блок, і 10-20 секунд для Ethereum. Відповідно узгодження проходять значно швидше. Незважаючи на всі переваги швидкості роботи з’являється більша ймовірність помилки. І якщо половина комп'ютерів, що працюють на даному ланцюгу, записують помилку, вона згодом стає постійною, а створення більш швидких блоків означає, що їх опрацює менша кількість систем [15].

Використовуючи прозорі дані, блокчейн дає змогу створювати нові промислові процеси, що забезпечує надзвичайно швидку реалізацію транзакцій завдяки можливостям розумних контрактів.

Було проведено дослідження, в якому Euroclear визначила основні переваги блокчейну.

Серед них [16]:

- надійність методів шифрування;
- конфіденційність даних;
- в будь-який момент часу база даних забезпечує надійність роботи без центрального органу;
- на противагу традиційному контракту, розумний контракт є алгоритмом, що програмується в залежності від умов, що несе в свою чергу виконання закладених інструкцій для роботи над різними процесами. Через це можна без проблем та витрат часу вдосконалити різні операції з цінними паперами, векселями і тд.;
- синхронізація даних між учасниками процесу;
- повніші набори даних;
- розподілена база даних дозволяє відійти від звичних традиційних централізованих систем, які використовуються в наш час для відстеження та ведення записів по різних інвестиціях і угодах;
- контрагенти почнуть більше довіряти один одному, отримавши доступ прозорий доступ до даних, не потрібно буде додатково узгоджувати і звіряти дані про угоди; відкритість даних також, дозволить учасникам підтверджувати свою надійність, збільшувати рівень довіри та цим самим зменшувати ризики фінансових забов’язень;
- швидше здійснення транзакцій. У всіх буде єдиний доступ до даних, що в свою чергу призведе до більш швидкого розповсюдження інформації на ринку.

2.4. Розумні контракти

З розвитком блокчейну велика кількість компаній цікавляться новими можливостями технології. Найбільш перспективним є використання розумних контрактів. Алгоритми, в основі яких лежить автоматизація комерйійних контрактів.

Щоб визначити, найефективнішу платформу для розумних контрактів та найбільш зручну для використання в цьому проекті, треба вивчити дану технологію якогомога детальніше.

2.4.1. Умови роботи смарт-контрактів

Смарт контракт являє собою спеціальну угоду, яка потрібна для укладання та здійснення комерційних контрактів, а транзакції між сторонами не вимагають участі сторонніх осіб [17]. Такі угоди автоматично виконують усі інструкції контракту і проводять санкції за порушення умов сторонами, вміщаючи в себе дані про зону відповідальності сторін.

В 1994 році вчений Нік Сабо перший раз висунув концепцію інтелектуальних контрактів. Він визначив, що розподілена база даних, може лягти за основу розумних угод. Ці контракти мають можливість перетворитися в програмний код, зберігатися, копіюватися, оброблятися та контролюються в системі комп'ютерної мережі, до якої застосовано блокчейн. Нею почали корситуватись на практиці аж за чотирнадцять років після винайдення, завдяки

тому, що з’явилась технологія блокчейн. Відповідно в алгоритму біткоїна було закладено принцип дії інтелектуальних контрактів, та на превеликий жаль так і не були використані в програмному забезпеченні через питання безпеки [17].

Тільки після запуску платформи Ethereum в 2015 році розумні угоди нарешті почали широко зостосовуватись. В ці дні саме цей ця платформа стала найкращою для розробки і роузмних контрактів, і децентралізованих додатків.

Умови, що обов’язково повинні виконуватись в смарт-контрактах [17]:

- розподілена база даних – платформа, на основі якої виконуватиметься смарт-контракт;
- учасники договору з електронними ідентифікаторами – сторони договору, що підтверджують умови договору;
- предмет угоди – об'єкт, всередині самого смарт-контракту. Це, наприклад, цифрові валюти, що гарантують прямий доступ контракту до предмету угоди без людського фактору;
- інструкції – алгоритм, що забезпечує коректне виконання всіх частин предмета договору програмною реалізацією.

Смарт-контракти програмуються на мові Solidity, яка була розроблена безпосередньо для виконання такої задачі. Ця мова зараз використовується переважно на платформі Ефіріум.

На Ethereum Virtual Machine можливо розробити програмний код для усіх скінченних алгоримів, так як EVM підтримує Тьюринг повноту.

Звичайно обмеження теж існують. Вони виникли через децентралізованість.

Є 2 типи акаунтів в Ethereum [18]:

- EOAs - externally owned account. Потрібно завантажити протокол у мережу протокол, створити приватний ключ і тоді створиться локальний обліковий запис;

- контракт. Має власний код, чим і відрізняється від попереднього акаунту. Існує можливість для коригування, керування та виклику інших контрактів.

Поєднання адрес гаманців і контрактів є ідеальним для зберігання Ethereum. Є можливість внутрішніх операцій, що означає, що контракти можуть співпрацювати з іншими. Тобто, ці внутрішні контракти створюються за рахунок зовнішніх рахунків.

Транзакції починають роботу контрактів (рис. 2.6).

```
eth.sendTransaction({from:
‘0xCBDB4E32a092203567487DAE4Abf66a7a7cbEE63’, to:
‘0xa8ade7feab1ece71446bed25fa0cf6745c19c3d5’, value: 1000000})
```

Рисунок 2.6 – Транзакція

Транзакція включає в себе вхідні параметри, які показують, як буде діяти контракт (мал. 2.7). Як зазначено в описі транзакціїї, вона містить [19]:

- одержувача;
- підпис, що ідентифікує учасника-відправника та погоджує ціль відправити інформацію через блокчейн;
- поле “VALUE”;
- поле даних, яке необов’язкове і може вміщувати в себе повідомлення;
- значення “STARTGAS” - максимальна к-сть кроків, що можлива для здійснення транзакцій;
- значення “GASPRICE”.

```
{

a transaction with 10 ether,
GASPRICE being 0.001 ether,
STARTGAS is 2000 gas,
the data field has 64 bytes.

}
```

Рисунок 2.7 – Вхідні параметри

Коли виконується контракт, існує можливість надсилання повідомлення на інший договір. Повідомлення включає в себе інформацію про відправника, одержувача, WEI, VALUE, додаткове необов'язкове поле даних та STARTGAS, яке обмежує кількість газу. Включає запуск коду, при необхідності [19].

Те що первинним є контаркт, а не екстернал акаунт і є різницею між повідомленням і транзакцією. Тому повідомлення для початку повинне запустити контракт-отримувач аби виконати свій власний код так само, як і транзакцію.

В Ethereum є криптовалюта, що називається саме ефіром, а не ефіріумом. Криптовалюта - це спосіб оплати, який розраховується за допомогою EVM.

Вартість цих розрахунків виражається в газі (і навіть не дивлячись на те, що фактична валютиа нестабільна, газ - постійна, що виражає ціну мережевого ресурсу).

Газ має статичне значення (вартість газу), і він завжди стабільний. Ціна вимірюється в криптовалюті (ціна на газ) і визначається відповідно до її коливання: при підвищенні ціни ефіру, ціна на газ повинна бути зменшена, щоб фактична вартість газу була завжди незмінною. Вартість газу, яка базується на обсязі та запиті, розраховується шляхом множення поточної ціни на газ, а загальна вартість відображається в криптовалюті.

Існує газовий ліміт, що включає в себе максимальне навантаження, обсяг операцій і розмір блоків. Майнер може повільно змінювати показники газового ліміту з часом.

Щоб виконати хоч одну транзакцію чи інструкцію, обов'зково вимагається сплачення суми газу майнерам. Звичайно, що є так званий список зафіксованих комісій за проведення конкретних операцій.

Газ оплачується лише в ефірі відповідно до метричної системи деномінацій Ethereum.

За цією системо усяка одиниця має своє назву.

Базовий блок називається Wei (рис. 2.8.) [20].

Unit	Wei Value	Wei
wei	1 wei	1
Kwei (babbage)	1e3 wei	1,000
Mwei (lovelace)	1e6 wei	1,000,000
Gwei (shannon)	1e9 wei	1,000,000,000
microether (szabo)	1e12 wei	1,000,000,000,000
milliether (finney)	1e15 wei	1,000,000,000,000,000
ether	1e18 wei	1,000,000,000,000,000,000

Рисунок 2.8 – Система деномінацій

2.4.2. Токени стандарту ERC20

Нещодавно «Initial coin offering» стала найпоширенішим способом збору коштів на реалізацію проектів в індустрії криптовалют. Основна частина таких

проектів розробляють самрт-контракти на платформі Ефіріум. Коли відбувається залучення коштів керівники проектів отримують біткоїни або ефір в обмін на токени, що вони начислюють інвесторам. Це так званий цифровий актив, що має якусь фінансову цінність. Токени зазвичай відповідають стандарту ERC-20. Що гарантує можливість роботи з децентралізованими додатками, так як вони теж притримуються такого стандарту, і дозволяє активам бути більш взаємозамінними.

ERC-20 визначає інструкції, які потрібно виконати, щоб токен прийняли до стандарту і здобув функцію взаємодіяти з іншими токенами [21].

2.4.3. Смарт-контракти на практиці

Розумні контракти значно полегшують роботу в різних областях, покращують відносини ділових партнерів за рахунок відкритості, контролюють рівень безпеки та значно покращують фінансові показники, зменшуючи багато витратат.

Наприклад, було проведено дослідження, що показало, що банківська галузь може заощадити до дванадцяти мільярдів доларів на рік, за рахунок використання новітніх криптотехнологій. Якщо розробити виборчу систему на основі смарт-контрактів, буде нарешті подалано порушення у даній сфері, наприклад, під час голосування, у вигляді підробок результатів або інших маніпуляцій. В логістиці система зможе значно заощадити час та відслідковувати роботу онлайн, бо не доведеться узгоджувати всім учасникам процесу свої дії, так як це вимагається зараз. В підприємництві, використання інтелектуальних контрактів дозволить здійснювати платежі працівникам та клієнтам автоматично, якщо будуть виконані попередньо закладені умови[23].

2.4.4. Недоліки смарт-контрактів

Є і мінуси в розумних контарктах, а саме [24]:

- Ціна і складність розробки та впровадження системи в роботу компанії. Для стабільності і коректності роботи потрібні кваліфіковані розробники, яких по-перше складно знайти, по-друге – вони дорого коштують;
- законодавство – в багатьох країнах закон не визнає і не визначає статус криптовалют;
- людина – алгоритм смарт-контракту є надзвичайно складним, потрібно прорахувати всі ймовірні варіанти розвитку, умови тощо. Цим займається людина, через що ризик помилок неймовірно високий. Якщо потрібно автоматизувати складний процес, відповідно настільки складнішим і стає алгоритм, що веде за собою збільшення ймовірності винекнення помилок.

2.5. Proof of Work, Proof of Stake

Якщо у традиційних мережах юзери не завжди можуть зв’язуватись напряму і часто повинні залучати ще когось аби затвердити якісь угоди, то блокчейн дозволяє їм не залучати іншу особу, а дані про транзакції вносити в розподілену базу даних, що доступна для всіх. Для того, щоб зробити це можливим, потрібно виконати умову консенсусу, а саме створити певний алгоритм підтвердження транзакції.

Peer to peer мережам, як і іншим децентралізованим механізмам, потрібно розв’язувати важку математичну проблему - вирішення конфлікту.

Буває так, що 2 факти, що були надіслані приблизно в той самий проміжок часу, можуть прийти в різні вузли і іншому порядку. А мережі треба вирішити, який факт прийшов спочатку[25].

Для досягнення консенсусу використовуються запрограмовані спеціальні алгоритми, і найпоширенішими є PoS і PoW.

2.5.1. Proof of Work

У випадку з Proof of Work, система вимагає від юзера розв’язання математичної задачі. Вона має бути складною для розв’язання, і в той же час але легкою для перевірки. В таких криптовалютах як біткоїн та лайткойн, алгоритм PoW існує у певній формі вузлів, що є конкурентами один з одним, а блок повинен бути прийнятий в блокчейн такої системи тільки одним з них. За допомогою криптографії і досягається такий ефект.

Для того, щоб розв’язати задачу потрібно знайти хеш, тобто одне вірне значення серед великої кількості. Для цього потрібно використовувати потужне комп’ютерне обладнання. Таке значення намагаються віднайти безліч майнерів і тільки той хто знаходить його першим і отримує винагороду, а той хто не встиг повинен починати все спочатку. І чим потужніше і краще обладнання, тим більша існує ймовірність першому знайти хеш [25].

2.5.2. Алгоритм Proof of Stake

Цей алгоритм є ще принципово іншим способом досягнення консенсусу та перевірки транзакцій. Мета в даному випадку схожа на PoW, але умова консенсусу тут кардинально відрізняється.

У випадку PoS творець нового блоку вибирається не шляхом гонки на вирішення математичних задач, а детерміністично, в залежності від кілскості токенів (цифрових асетів), якими він володіє. Плата за видобутий блок тут відсутня, а хешрейт мережі підтримуються за рахунок плати за транзакції.

Алгоритм Proof of Stake працює таким чином [26]:

а) користувач купує валюту, що працює за алгоритмом PoS;

б) завантажує програму-майнер та запускає її в режимі форжинг та синхронізовує ланцюг;

в) очікує синхронізації та запускає майнинг;

Кошти почнуть надходити на вказаний гаманець впродовж 30 діб.

2.6. Мова програмування Solidity

Solidity – JavaScript-подібна мова програмування для розробки розумних контрактів для EVM. Програми на Solidity (Рис. 2.9.) транслюються в байт-код EVM. Solidity дозволяє розробникам розробляти програми, які містять логіку для створення алгоритмів проведення транзакцій в мережі Ethereum [27].

Мова була створена групою програмістів під керівництвом Крістіана Ратвізнера в 2014 році на основі ідей Кевіна Вуда. Подібність в синтаксисі

JavaScript - створена для швидкого опанування синтаксису нової мови розробниками.

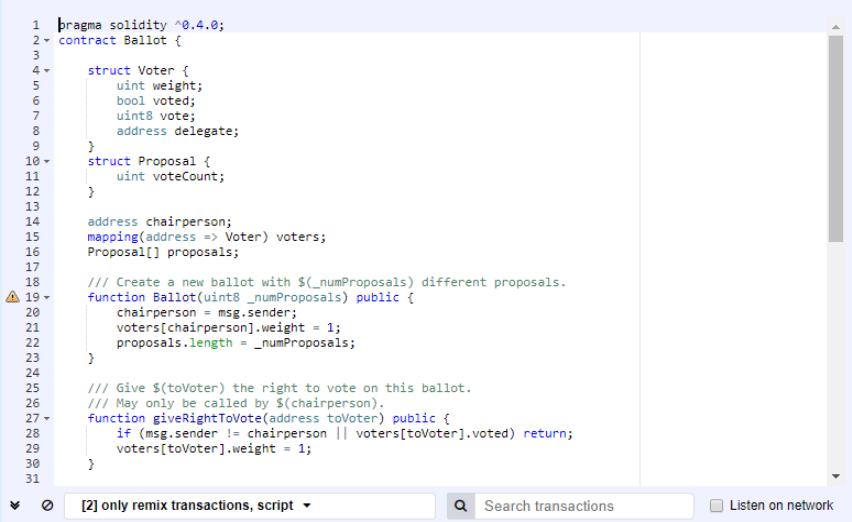

Рисунок 2.9 – Приклад програми мовою Solidity

На даний момент повноцінного релізу не було (поточна версія – 0.5.12), тому деякі особливості мови поки існують в урізаному вигляді, але незважаючи на це зі своїми задачами Solidity справляється. Solidity доступний для великого числа IDE і редакторів:

- IntelliJ IDEA ;
- Microsoft Visual Studio ;
- Atom Linker.

Solidity має наступні особливості [28]:

а) статично типізований. Це викликано необхідністю аналізу змінних перед запуском контракту, а також для передачі змінних в параметри функцій;

б) Розширений набір ідентифікаторів доступу:

- public – може бути викликана ззовні контракту;
- private –може бути викликана тільки всередині контракту;
- view – не змінює стан, а зчитує деякі дані(аналог геттерів);
- pure – функція відповідає принципам функціонального програмування без сторонніх ефектів;

- internal – доступний в контрактах - наслідниках (аналог protected);
- external – може бути викликана тільки ззовні контракту;

Наприклад, функція, зчитюча дані, що доступна ззовні буде мати наступні ідентифікатори:

```
function getUserId (string _name) public view {
// function code goes here
}
```

в) змінні також мають 2 можливих ідентифікатора доступу:

- storage – змінні, які зберігаються вBlockchain ;
- memory – тимчасові змінні.

Важливою деталлю є те, що якщо оголошено структуру або масив у межах функції, необхідно чітко вказати, що ця змінна повинна зберігатись у Storage або в Memory. Приклад:

```
User storage myUser = user[_userId];
```

Тут ініціалізується структура типу User з ім’ям myUser з масиву по індексу _userId, зберігаючи інформацію в Blockchain;

г) Щоб порівняти значення двох змінних, необхідно користуватися алгоритмом хешування, наприклад keccak256. Приклад:

```
(keccak256(_name) == keccak256("Matt"))
```

г) При виклику функції ззовні, часто необхідно знати адресу гаманця, з якого ініційовано виконання даної функції. Msg.sender – глобальна змінна, значення якої може бути отримано в тілі функції;

д) mapping є асоціативним масивом в Solidity, що зберігає інформацію за принципом ключ-значення. Наприклад:

```
mapping (uint => address) public postToOwner;
postToOwner[id] = msg.sender;
```

е) За допомогою ключа “require” ми можемо перевірти обов’язкову умову до продовження виконання функції.

```
function deletePost(uint _id) public {
    require(msg.sender == postToOwner[_id]);
    // delete post
}
```

2.7 Веб-інтрефейс до смарт-контракту

Найважливішою складовою веб-інтерфейсу є його взаємодія з безпосередньо смарт-контрактом.

Існує кілька бібліотек, які реалізують злагоджену роботу і веб-додатків та смарт-контрактів. Прикладом такого програмного модулю є Web3.js, який на думку багатьох експертів і користувачів ресурсу StackOverflow є на данний момент найбільш пристосованим до роботи в реальних умовах.

Web3.js - Бібліотека JavaScript для взаємодії з Ethereum смарт-контрактами, а точніше контрактними блоками [29].

Бібліотека має кілька основних цілей:

- Ініціювати виконання функції смарт-контракту;
- Моніторинг подій, які виникають при виконання функцій;

Існує кілька способів побудови взаємодій між веб-додатками та розумними контрактами.

Перша - завантажити Web3 в браузер, щоб у подальшому працювати з ним безпосередньо з клієнтської сторінки. Хоча цей підхід є не цілком правильним, оскільки це відкриває інтерфейс RPC, що не є цілком безпечно.

Іншим способом взаємодії додатків та контрактів є написання server-side бібліотеки-обгортки, яка реалізує підключення до вузла Ethereum для безпосереднього зв'язку з розумним контрактом.

Цікавим моментом є те, що готові бібліотеки-обгортки вже є у відкритому доступі і можуть бути реалізовані не тільки мовою JavaScript, що дозволяє мультимовну взаємодію та значно розширює стек технологій, можливих до використання в блокчейн-проектах [29].

Особливий інтерес для нас представляє Web3Scala, що дозволяє нам користуватись усіма функціями даної бібліотеки без надлишкових дій.

Тепер розглянемо детальніше можливості бібліотеки Web3.js, код для якої є по-суті псевдокодом для використання цієї бібліотеки з будь-яким компілятором.

Після підключення, потрібно створити екземпляр класу для установки з’єднання:

```
if (typeof web3 !== 'undefined') {
  web3 = new Web3(web3.currentProvider);
} else {
  // set the provider you want from Web3 providers
  web3 = new Web3(new
Web3.providers.HttpProvider("http://localhost:8495"));
```

```
}
```

Методи цієї бібліотеки дозволяють відстежувати події. Подія Ethereum має безліч призначень:

- отримати значення, що повертають методи, що змінюють стан смарт-контракту;
- отримання повної історії дій над смарт-контрактом.

Наприклад, тут ми отримуємо інформацію про контракт і реальному часі за допомогою інтерфейсу програми allEvents():

```
var events = corecontractContractInst.allEvents();
// watching for all changes
events.watch(function(er, ev){
if (!er)
console.log(JSON.stringify(ev));
});
```

Крім цих прикладів, що наведені вище, цей програмний модуль має величезну кількість можливих сценаріїв застосунку.

Висновки за розділом 2

В результаті аналізу, проведеного в даному розділі, ми отримали чітке розуміння про технологої, які будуть використані в плануванні, реалізації та тестуванні програмого продукту.

Середовище виконання смарт-контракту буде приватним блокчейном, що буде запущений на віртуальному сервері. Мовою програмування для реалізації смарт-контракту було вибрано Solidity, а середовищем тестування є ethereum-гаманець Mist.

Для розробки було вибрано мову програмування Scala, а робота по протоколу http буде реалізована за допомогою Play! Framework.

Для організації взаємодії між смарт-контрактом і веб-додатком буде використана бібліотека Web2Scala, що є обгорткою до описаної вище бібліотеки Web3.js.

3 АНАЛІЗ РОБОТИ ПРОДУКТУ

3.1 Приватний блокчейн на базі Ethereum

У попередньому розділі був даний огляд різних середовищ роботи смарт-контрактів. Проаналізувавши їх, ми вирішили вибрати платформу Ethereum.

Ethereum такий популярний через велику кількість клієнтів (деякі з яких - термінальні, частина - мають графічний інтерфейс та декілька гібридних рішень). Основним є geth, створений командою Ethereum.

Geth (Go Ethereum) - це одна з реалізацій вузла Ethereum. Він був написаний на Go (мова програмування, розроблена Google) і має повністю відкритий код та ліцензію GNU LGPL v3. Go Ethereum можна завантажити як автономний кросплатформенний клієнт, або як програмний модуль мов програмування Go, Java або Swift [30].

У майбутньому всі дії будуть розглянутів контексті операційної системи Mac OS X.

Щоб встановити geth, має бути попередньо встановлений homebrew.

Наступним кроком є створення облікового запису для ресурсу Ethereum. Це дозволить створити деякі приватні та відкриті ключі та паролі для захисту.

Різні ланцюги Ethereum відрізняються один від одного початковими блоками. Кожен блокчейн починається з такого блоку, а всі інші блоки додаються поверх нього, при цьому кожен наступний містить в собі хеш-посилання на попередній. Первинний блок основної ланки Ethereum був доданий 20 липня 2015 року. Для приватного блокчейну також потрібно створити основний блок. Для виконання ініціалізації виділеного Blockchain з заданим первинним блоком використовується така команда:

```
geth –datadir <path-to-data-directory> init <path-to-genesis-block>,
```

де:

- datadir: директорія для зберігання інформації і ключів;
- init: шлях до нового genesis блоку, представленого файлом json.

Нижче продемонстрований genesis (первинний) блок, який буде першим в нашому ланцюгу (рис.3.1).

```
{
    "config": {
        "chainId": 15,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "difficulty": "0x400",
    "gasLimit": "0x2100000",
    "alloc": {
        "7a69b359e86893efa3d9732e4c65ced51567edd0":
         { "balance": "0x1337000000000000000000" }
    }
}
```

Рисунок 3.1 – Genesis блок в нашому Blockchain

Після того, як це зроблено, можна почати майнинг. Networkid є унікальним ідентифікатором цього ланцюжка. Після цього всі учасники, які бажають використовувати даний блокчейн, повинні використовувати той самий мережевий ідентифікатор з тим же первинним блоком.

```
Geth -mine -rpc -networkid <networkd-id> --datadir
<path-to-data-directory>
```

Існує два варіанти запуску майнингу. Запустити консоль власноруч або під’єднатись до хосту за допомогою команди attach.

Консоль підключиться до вузла за протоколом IPC. IPC (Interprocessor Interaction) працює локально. В такому разі, geth створює транспорт в файловій системі (представлений файлом <path-to-data-directory> /geth.ipc) - консоль підключится до хосту за допомогою IPC.

3.2 Mist Wallet

Ця система не тільки зберігає монети Ethereum, але також дозволяє їх відправляти та отримувати. Існує ще одна важлива особливість даного гаманця - це можливість створювати та виконувати розумні контракти.

Є три типи мереж [29]:

- основна мережа - відкрита для всіх, і всі транзакції Ethereum проходять через неї;
- тестова мережа для імітації основної мережі при розробці розумних контрактів, для того щоб зберегти фактичні ресурси. Реальної вартості грошей або транзакцій, що відбуваються в мережі - немає;
- приватні мережі. Можна створити власний приватний Blockchain або підключитися до існуючого ланцюга. Як правило, ці мережі призначені лише для тестування.

Коли ви запускаєте гаманець Mist, він з'єднується з основною мережею за замовчанням. Після запуску Mist обліковий запис користувача стає частиною блокчейну, що означає, що він має мати копію всього ланцюга. При запуску гаманець автоматично синхронізується з Blockchain. Це може зайняти до кількох днів. Отже, спочатку доцільно запустити Mist в приватній мережі. Ви

можете вибрати тестову мережу в стартовому вікні гаманця. Для завантаження не потрібно завантажувати таку кількість даних, але для синхронізації з тестовою мережею все ще потрібний деякий час.

На верхньому краю інтерфейсу можна побачити строку статусу синхронізації. Якщо вузол перестав синхронізуватись (наприклад, нульові вузли чи висота блоків залишилась постійною), потрібно перезапустити програму Mist або зачекати.

У розділі "Облікові записи" ви можете переглянути створені облікові записи та баланс Ethereum у кожному з них (Рис. 3.2).

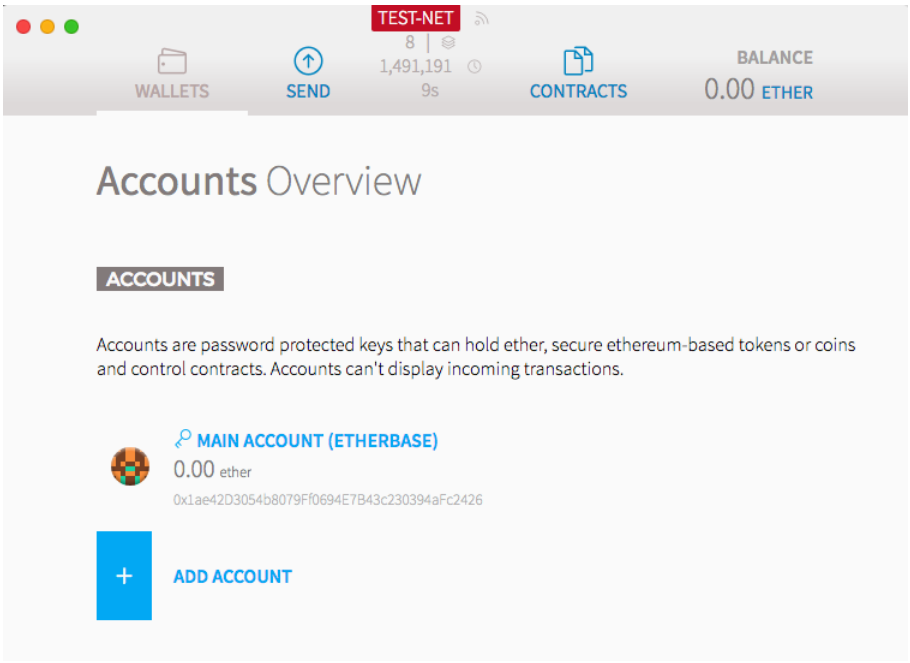

Рисунок 3.2 – Стартове вікно Mist

Для подальшого використання розумних контрактів необхідно додати новий обліковий запис, який може використовувати розумні контракти. Якщо ви увійшли в систему, в гаманці вже є основний обліковий запис з унікальною адресою, а щоб додати обліковий запис вам потрібно буде натиснути “Додати обліковий запис”, заповнити всі поля та створити нову унікальну адресу для облікового запису.

3.3 Налаштування вузла

Вузол - це комп'ютер, що є складовою частиною Blockchain мережі. Вузол розташований в децентралізованій мережі, використовує протокол P2P для підключення та обміну інформацією про блоки та транзакції [30].

Щоб розпочати, введіть в термінал "geth". Ця команда запустить працювати вузол в основному Blockchain і розпочне синхронізацію блоку. Оскільки вирішено не проводити синхронізацію всіх блоків, вам слід написати:

```
geth –fast –cache 1024.
```

Щоб синхронізація пройшла швидше, ви можете завантажувати тільки заголовки, що прискорить процедуру. Для цього вкажіть прапорець -fast.

Основним варіантом є консоль, яка дозволяє вам використовувати інтерактивний режим geth. Щоб запустити Geth в режимі Private Blockchain, вам слід використовувати прапорець:

```
geth –dev console.
```

3.4. Реалізований смарт-контракт так опис його роботи

3.4.1. Опис архітектури

Контракти Solidity є аналогією з класів у широкому значенні, але їх задача цілком відрізняється від задачі класів у більшості популярних мов

програмування. Проте контракти містять поля, методи та стандартні (приватні, загальнодоступні) ідентифікатори доступу.

Розроблений розумний контракт призначається для основних функцій перерозподілу криптовалютних активів, тому в нашому випадку функції смарт-контрактів є:

- створення вхідних даних про власників та кількість їх акцій;
- додавання нового власника за згодою вже існуючих акціонерів та виділення йому певної кількості дольових забов’язань (рішення вважається прийнятим, якщо підтримане більш ніж 50% акцій);
- додавання нового власника та передача йому активів іншого власника за його згоди (продаж долі);
- функція виплати дивідендів;
- переказ монет з корпоративного гаманця на задану адресу за згодою акціонерів;
- ліквідація компанії.

Для компаній, які використовують державну валюту для розрахунків, кількість дій, які може виконати рада директорів є набагато більшою, але всі ці можливі дії можуть бути записані як сукупність наведених вище дій. Однак відповідно до сформульованого нами технічного завдання, код розумного контракту матиме можливість збільшити можливі функції акціонерів шляхом додавання нових частин програми та зміни вже існуючих.

Для інкапсуляції інформації і дій акціонера реалізована структура Shareholder (Рис. 3.3), яка включає в себе:

- ім'я;
- опис
- адреса гаманця;
- кількість акцій, що йому належать.

```
struct Shareholder {
  bytes32 name; // Повне ім'я акціонера
  bytes32 description; //Додаткова інформація - посада, паспортні дані тощо.
  address account; // Адреса персонального гаманця.
  uint share; // Цілочислений тип - кількість акцій акціонера
}
```

Рисунок 3.3 – Shareholder

Абсолютна кількість акцій співвласинка використовується для того, щоб обійти обмеження в Solidity використання нецілочисельних типів.

3.4.2. Події (Events) та їх застосування в системі

Крім прямої функції перерозподілу коштів, розумний котракт виконує функції аудиту. Наслідками будь-якої дії підприємства, окрім безпосередньої функції, є емітування повідомлень (Event), що містять інформацію, які дії відбулися, і всі параметри та результати функції. Це призначено для обліку дій акціонерів, інвесторів та фінансових служб.

Solidity забезпечує можливість для створення подій, тому завдання розробника полягає лише в описі структури цих повідомлень та визначення часу та умов, за яких їх буде трансльовано в Blockchain.

Було розроблено набір подій, кожна з яких відповідає безпосередній дії підприємства (Рис. 3.4 - 3.5). Подій перерахування коштів тут немає, оскільки факт оплати зафіксований в Blockchain і не вимагає жодних додаткових доказів.

```
// Івент, що емітується при ініціалізації смарт-контракту
event AssociationCreated(
  bytes32[] shareholderNames, // Імена всіх співвласників
  uint[] shareholderShares, // Кількість акцій всіх співвласників
  address account // адреса рахунку підприємства
);

//Івент, що емітується після того, як користувач проголосував за пропозицію
event GotVote(
  uint voteProposalId, //Ідентифікатор пропозиції
  address voter, // Адреса користувача
  bool success, // Голос відданий успішно/не успішно
  bytes32 errorMsg // Повідомлення про помилку
);

//Івент, що емітується після того, голосування завершено
event VotingResult(
  uint voteProposalId, //Ідентифікатор пропозиції
  bool success // Рішення прийнято/не прийнято
);

//Івент, що емітується після того, як був додоаний новий акціонер
event ShareholderCreated(
  bytes32 shareholderName, // Повне ім'я акціонера
  uint shareholderShare, // кількість акцій акціонера
  address account // адреса персоанльного рахунку акціонаера
);
```

Рисунок 3.4 – Оголошення подій

```
//Івент, що емітується після того, як були виплачені дивіденди
event WithdrawFinished(
  bool success, // Розподілення дивідендів успішно/не успішно
  uint amount // Загальна сума дивідендів
);

//Івент, що емітується після того, як акції були переказані
//(Перейшли від одного акціонера до іншого)
event SharesTransfered(
  address from, // Адреса того, хто передав
  address to, // Адреса того, кому передав
  uint amount, // Загальна сума переказаних акцій
  bool success, // Переказ успішно/не успішно
  bytes32 errorMsg // Повідомлення про помилку
);

//Івент, що емітується після того, як кошти з корпоративного рахунку
//були переказані
event TransferResult(
  address to, // Адреса того, кому переказано
  uint amount, // Загальна сума переказаних коштів
  bool success, // Переказ успішно/не успішно
  bytes32 errorMsg // Повідомлення про помилку
);

event NewVoting(
  uint id,
  bytes description
);

//Івент, що емітується перед тим як контракт самоліквідується
event ContractDestroyed();
```

Рисунок 3.5 – Оголошення подій

3.4.3. Ініціалізація смарт-контракту

Під час створення смарт-контракту Blockchain розміщює згенерований код, який містить інформацію про акціонерів компанії та адреси їх електронних гаманців. Ці дані формують масив stakeholders що є повним списком усіх співвласників підприємства.

З наведеного коду ви можете побачити, що при створенні контракт емітує подію AssosiationCreated, яка свідчить про те, розумний контракт компанії успшно створений. У цьому випадку ви можете отримати інформацію про акціонерів та кількість їхніх акцій, а також адресу облікового запису компанії (Рис. 3.6).

```
// Конструктор, який виконується при початковій ініціалізації контракту
function Association() public payable {
  // Згенерований код ініціалізації початкових акціонерів
  shareholders.push(Shareholder("Lupa", "CEO", 0x1EB65388000401CE2AF05b058a150A78d51c170e, 100));
  shareholders.push(Shareholder("Pupa", "CTO", 0xB9b8e6b85C8f23859b1146c6E20A58225D2C3747, 50));

  // Збір інформації та емітування івенту про створення
  for (uint i = 0; i < shareholders.length; i++) {
    shareholderNames.push(shareholders[i].name);
    shareholderShares.push(shareholders[i].share);
  }

  emit AssociationCreated(shareholderNames, shareholderShares, address(this));
}
```

Рисунок 3.6 – Конструктор

3.4.4. Рада директорів

При додатковій емісії акцій, виплаті дивідендів, ліквідації, а також коли кошти перераховуються на сторонні рахунки, виникають пропозиції, які мають бути підтримані більшістю співвласників, тому смарт-контракт повинен передбачати голосування. Це означає, що лише тоді, коли "за" рішення про участь в угоді проголосуть 50% акцій вона може бути прийнята до реалізації.

В системі ця функціональність реалізується за допомогою структури VoteProposal, яка включає в себе дані питань на порядку денному (див. Рис. 3.7). Час UNIX (Пройдена кількість мілісекунд з 1 січня 1970 р.), є ідентифікатором кожного питання, крім цього структура містить інформацію

про акціонера, що виставив питання на голосування, статус та опис угоди на голосуванні.

```
struct VoteProposal {
  uint id; //Ідентифікатор пропозиції для голосування
  uint _type; // Тип операції, яка буде виконана після завершення голосування
  address creator;
  uint sharesVotesCount; // Кількість голосів "за"
  bool status; //Статус голосування (true - відкрите, false - закрите)

  //Додаткова інформація про голосування
  address addr;
  bytes32 name;
  bytes32 descr;
  uint share;
}
```

Рисунок 3.7 – Структура VoteProposal

Коли один з користувачів активує одну з функцій, розпочнеться процес голосування:

- addNewShareholderProposal - призначити нових акціонерів із зазначеною кількістю акцій;
- withdrawProposal - випуск дивідендів;
- destroyContractProposal - закінчення діяльності смарт-контракту;
- transferProposal - передача коштів третім особам;

Після висунення питання, Blockchain отримує сигнал про подію NewVoting, в якому міститься опис ідентифікатора пропозиції та самого питання. Для прикладу можна звернути увагу на addNewShareholderProposal (Рис. 3.8). Серед параметрів функції є всі змінні, які описують можливих нових співвласників, а тіло функції лише додає питання до наявного списку голосувань та надає йому ідентифікатор для подальшого голосування. В кінці всіх вищезазначених функцій в Blockchain відправляється подія про початок голосування.

```
function addNewShareholderProposal(bytes32 name, bytes32 description, address account, uint share) public {
  uint id = now;
  voteProposals.push(
    VoteProposal(id, 0, msg.sender, 0, true, account, name, description, share)
  );
  //Емітування сигналу про початок голосування
  emit NewVoting(id, msg.data);

}
```

Рисунок 3.8 – Функція addNewShareholderProposal

Ті, хто підтримує цю ініціативу, ініціюють функцію vote, вказуючи в ідентифікатор голосування, взятого з сигналу про початок голосування, таким чином додаючи кількість своїх акцій в лічильник акцій “за” ініціативу. (Рис. 3.9).

```
function vote(uint id) public {
  bool voteSuccess = false;
  bytes32 errMessage = "Vote ID is incorrect";
  for (uint i = 0; i < voteProposals.length; i++) {
    //Пошук питання що подано на розгляд
    if (voteProposals[i].id == id && voteProposals[i].status) {
      errMessage = "Cannot find voter";
      for (uint j = 0; j < shareholders.length; j++) {
        //Пошук інформації про людину, що голосує
        if (shareholders[j].account == msg.sender) {
          //Додання її акцій до тих, що виступають "за"
          voteProposals[i].sharesVotesCount += shareholders[j].share;
          voteSuccess = true;
          errMessage = "";
        }
        break;
      }
      break;
    }
  }
  //Емітування сигналу про те, що користувач успішно/неуспішно проголосував
  emit GotVote(id, msg.sender, voteSuccess, errMessage);
}
```

Рисунок 3.9 – Функція vote

В кінці процесу голосування, акціонер, що виставив питання порядок денний оголошує результати функцією finishVote. Результатом виконання цієї функції є емітування сигналу про неуспішне голосування або ж приведення

угоди в дію. У другому випадку виконується одна з 4 функцій, наведених нижче:

- addNewShareholder - додавання нового акціонеріа;
- withdraw - виплата дивідендів;
- destroyContract - ліквідація смарт-контракту;
- transferProposal - передача коштів на рахунок.

Тепер розглянемо функцію, що відповідає тій, що ми описували вище AddNewShareholder (Рис. 3.10) Дана функція захищена від виконання вручну приватним ідентифікатором досупу, тому вона можлива до виконання тільки після голосування. Під час нього параметри виклику функції зберігаються в структурі питання на голосування. Робота функції аналогічна до конструктора.

```
function addNewShareholder(bytes32 name, bytes32 description, address account, uint share) private {
  shareholders.push(Shareholder(name, description, account, share));
  //Емітування сигналу про те, що додано нового акціонера
  emit ShareholderCreated(name, share, account);
}
```

Рисунок 3.10 – Функція AddNewShareholder

Аналогічно виконуються інші фунції, які передбачають голосування.

3.4.5. Функції смарт-контракту і їх взаємодія з Blockchain

Вище була детально досліджена функція addNewShareholder з точки зору виконання. Тепер давайте подивимося на всі інші випадки виконання функцій та емітувань подій на Blockchain.

Функція виплати дивідендів (Рис. 3.11) призначена для виплати сумірної кількості дивідендів що дорівнює сумі, яка передається як параметр. По-перше, перевіряється достатність коштів на балансі контракту. Далі шляхом

ітерування по масиву акціонерів кожному виплачується доля цієї суми відповідно до їх долі в компанії. Якщо не виникає ніяких помилок, то в Blockchain оголошується подія успішної виплати дивідендів з вказанням загальної суми виплат.

```
function withdraw(uint amount) private {
  //Перевірка можливості розподілу (Чи вистачає коштів)
  if (amount < address(this).balance) {
    uint totalShares = getTotalShares();
    for (uint i = 0; i < shareholders.length; i++) {
      //Розподіл коштів в залежності від частини акцій, якими володіє акціонер
      shareholders[i].account.transfer(amount * shareholders[i].share / totalShares);
    }
    //Емітування сигналу про успішний розподіл дивідендів
    emit WithdrawFinished(true, amount);
  } else {
    //Емітування сигналу про неуспішний розподіл дивідендів
    emit WithdrawFinished(false, amount);
  }
}
```

Рисунок 3.11 – Функція withdraw

Функція transfer призначена для переказу грошей на особистий гаманець одного з учасників товариства або ж на гаманець третьої сторони. В параметрах міститься адреса криптовалютного рахунку та загальна сума переказу (Рисунок 3.12). Спочатку перевіряється кількість коштів на балансі. Якщо у підприємства достатньо коштів для проведення транзакції, то виконується переказ і емітується сигнал transferResult з фактичним результаом та загальною кількістю переданих монет.

```
function transfer(address addr, uint amount) private {
  //Перевірка можливості здійснення переказу
  if (amount < address(this).balance) {
    //Переказ
    addr.transfer(amount);
    //Емітування сигналу про успішний переказ коштів третій особі
    emit TransferResult(addr, amount, true, "");
  } else {
    //Емітування сигналу про неуспішний переказ коштів третій особі
    emit TransferResult(addr, amount, false, "Insufficent funds");
  }
}
```

Рисунок 3.12 – Функція transfer

Функція transferShares (рис. 3.13) призначена для переказу дольових забовязань від одного акціонера до іншого. В параметрах міститься адреса криптовалютного рахунку реципієнта акцій та загальна кількість переказу. Спочатку перевіряється кількість акцій у донора та чи є він акціонером. Якщо всі умови виконуються, то наступний крок - знайти одержувача платежу. Якщо одержувача не знайдено, надається сигнал з негативним параметром та повідомлення "Не вдається знайти одержувача". Якщо одержувача знайдено, кошти переносяться і після успішного переказу надсилається сигнал, з позитивним параметром.

```
function transferShares(address to, uint amount) public {
  bool status = false;
  uint donorIdx;
  uint recepIdx;
  //Знаходження відправника
  for (uint i = 0; i < shareholders.length; i++) {
    if (shareholders[i].account == msg.sender && shareholders[i].share >= amount) {
      donorIdx = i;
      status = true;
    }
  }
  if (!status) {
    //Емітування сигналу про те, що в відправника немає достатньої кількості акцій
    emit SharesTransfered(msg.sender, to, amount, false, "Insufficent number of shares");
  } else {
    status = false;
    //Пошук отримувача
    for (i = 0; i < shareholders.length; i++) {
      if (shareholders[i].account == to) {
        recepIdx = i;
        status = true;
      }
    }
    if (!status) {
      //Емітування сигналу, що отримувача не знайдено
      emit SharesTransfered(msg.sender, to, amount, false, "Cannot find recepient");
    } else {
      //Переказ коштів
      shareholders[donorIdx].share -= amount;
      shareholders[recepIdx].share += amount;
      //Емітування сигналу про успішний переказ
      emit SharesTransfered(msg.sender, to, amount, true, "");
    }
  }
}
```

Рисунок 3.13 – Функція transferShares

3.5 Взаємодія смарт-контракту та його веб-інтерфейсу

Взаємодія веб-додатку та смарт-контракту відбувається через API бібліотеки Web3.js, яка доступна для Scala під назвою Web3Scala. Інтерфейс програмного забезпечення містить набір функцій, які взаємодіють безпосередньо з заданим вузлом Ethereum та дозволяють виконувати транзакції, включаючи створення смарт-контракту та виконання його методів.

Для створення смарт-контракту та його тестування, необхідно надати всю необхідну інформацію у формі веб-додатку. У цьому випадку передається вся інформація про співвласника підприємства, що впровадила смарт-контракт: імена акціонерів, їх позиція в компанії, кількість акцій і публічний ключ їх індивідуальних криптовалютних гаманців, які мають бути створені заздалегідь. Після цього веб-додаток за допомогою Web3Scala починає створення унікального смарт-контракту, записує його в реєстр Ethereum та запускає його до роботи. У Blockchain цей договір буде зберігатися в двійковому коді. Після цього створюється веб-сторінка, на якій відображається адреса контракту, яка буде використовуватися для виконання будь-яких операцій у майбутньому.

Але є ще один спосіб перевірити роботу розумних контрактів. Характерною рисою розвитку контрактної роботи є те, що ми можемо виконувати його без обов'язкового спеціального програмного забезпечення. Достатньо запустити гаманець, що працює зі смарт-контрактами Ethereum. В цьому проекті було прийнято рішення використовувати гаманець Mist,що буде клієнтом та geth, який виступить у ролі ноди.

Оскільки було припущено, що особа, яка використовує веб-додаток, не має навичок програмування, варто врахувати можливість вручну створити контракт в гаманці Mist.

Отже, щоб почати, вам потрібно підключитися до вузла через клієнт Mist, вказавши адресу сервера та порт, що хостить вузол (Рис. 3.14).

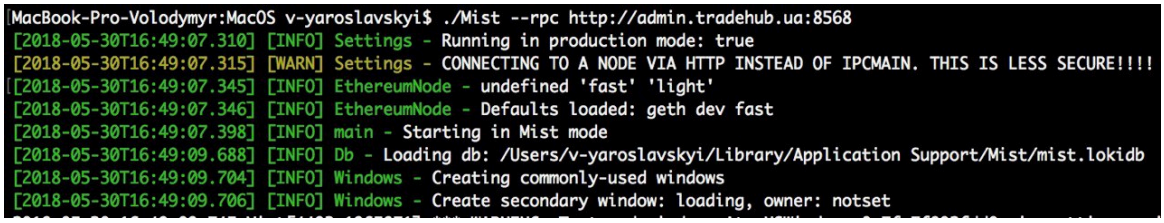

Рисунок 3.14 – Підключення вузла через консоль

Запустивши гаманець, бачимо інформацію про публічні ключі, створені раніше. В них вже є ефір, що був видобутий директивами geth (Рис. 3.15).

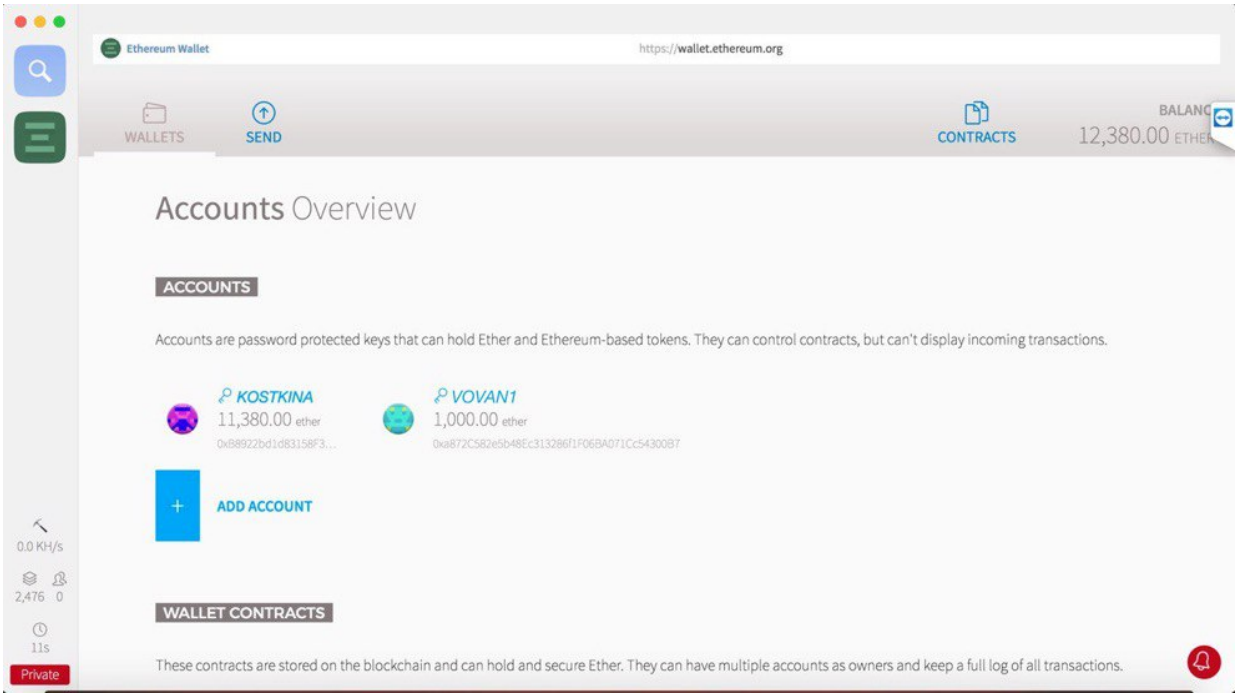

Рисунок 3.15 – Інтерфейс сторінки з аккаунтами

Наступним кроком є компіляція і занесення контракту в реєстр. Для цього треба вибрати "Contracts" у верхньому меню та вибрати "Deploy new contract"(рис. 3.16).

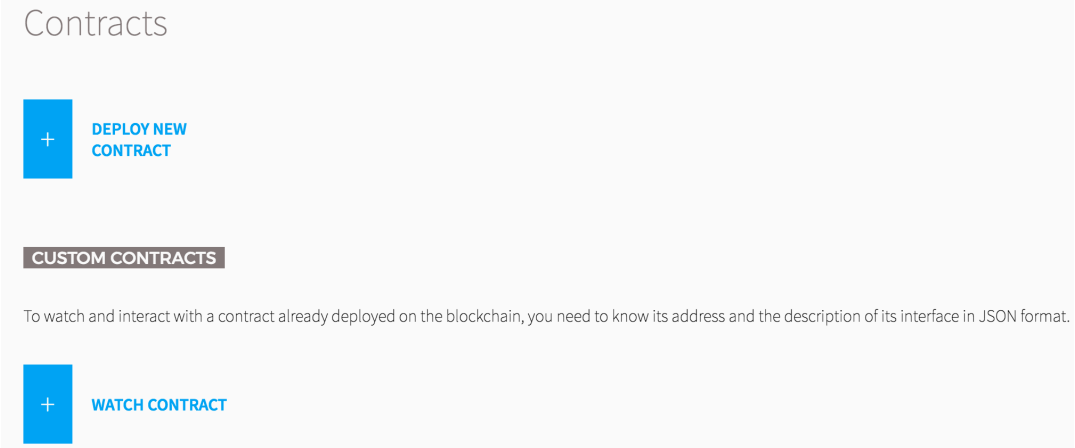

Рисунок 3.16 – Панель контрактів

Потім вам потрібно ввести адресу, від чийого імені ви створите контракт та деяку кількість криптовалюти, щоб виконати контракт. Це необхідно для достатнього постачання газом, тому що для кожна операція в смарт-контракті потребує коштів. Вартість транзакції визначається ціною газу.

Далі, у полі "Solidity contract source code" потрібно вставити попередньо написаний розумний контракт (рис. 3.17). Ви також можете налаштувати швидкість створення контракту. Але варто пам'ятати, що чим раніше контракт виконається, тим вище комісія.

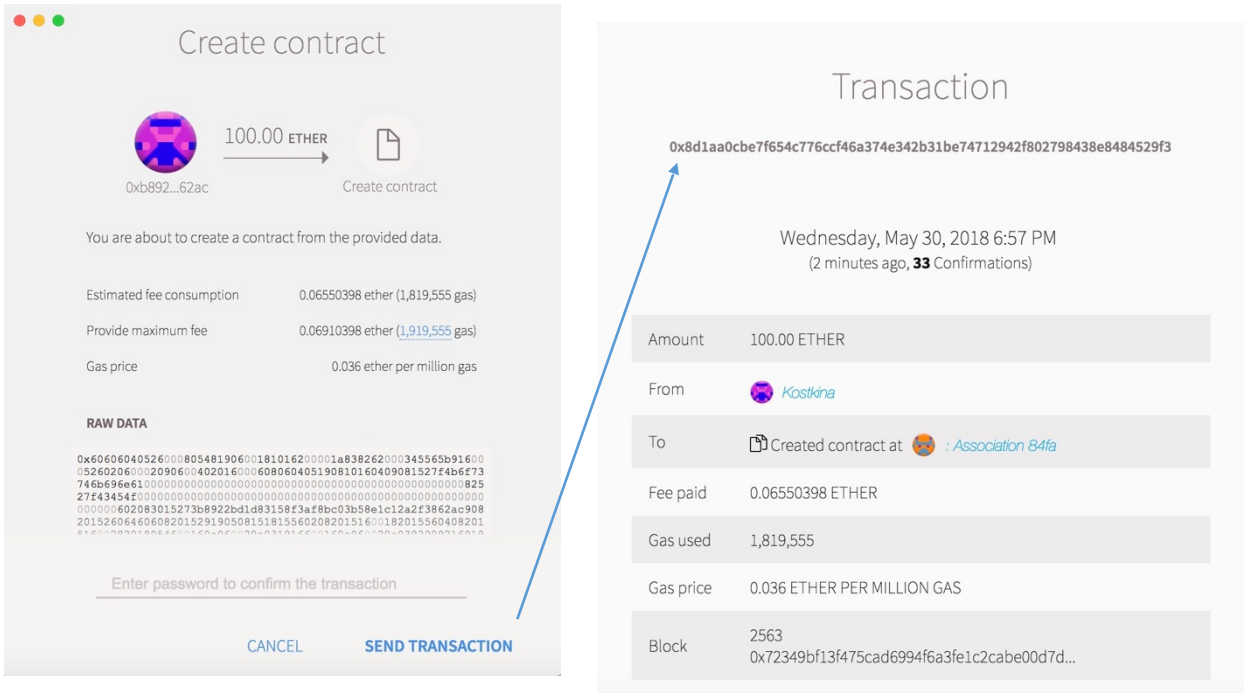

Рисунок 3.17 – Введення інформації по контракту

Потім з'явиться вікно з кінцевими даними та поле для введення пароля. Після введення ви можете переглянути всі дані транзакції, що отримали новий транзакційний ключ.(рис. 3.18).

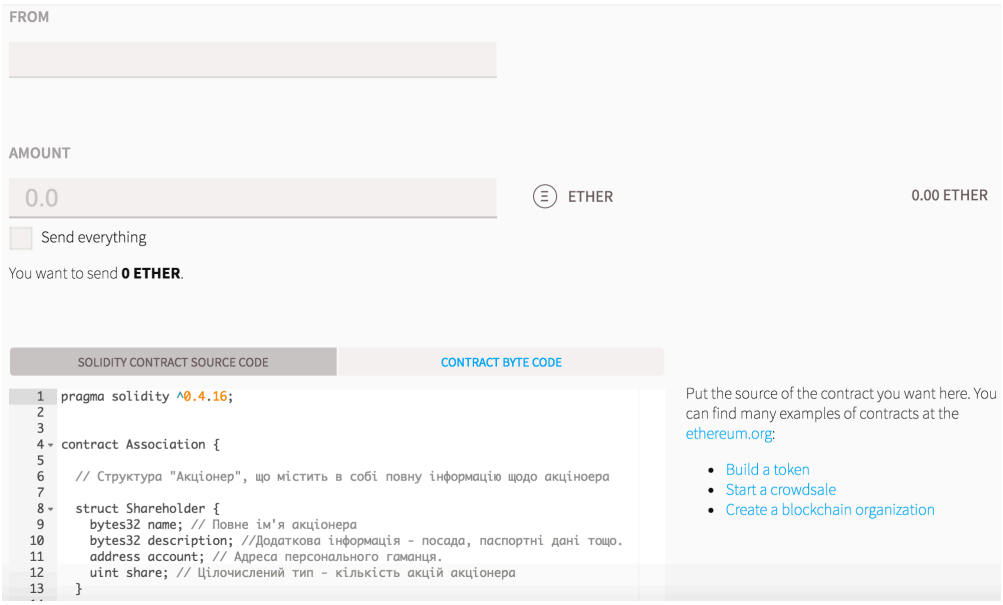

Рисунок 3.18 – Успішне створення контракту та інформація про пройдену транзакцію

Після ініціалізації контракту можна виконати всі описані в ньому функції. Щоб перевірити правильність його роботи, можна виконати деякі функції. Наприклад, "withdrawProposal" - це функція, яка ініціює розподіл дивідендів. Ви повинні вказати загальну суму дивідендів, що виплатяться та відправника. Потім натисніть кнопку "Виконати".

Після виклику цієї функції надсилається сигнал про початок голосування. Іншими словами, операція буде здійснюватися лише після того, як акціонери проголосують "за". Якщо більш ніж 50% акціонерів будуть позитивно голосувати, голосування буде успішним. Після успішного голосування дивіденди можуть бути передані (Рис. 3.19).

Сума дивідендів буде перерахована з попередньо створеного договору на корпоративний гаманець акціонерів і буде розподілятися відповідно до частки акцій в ньому.

Після голосування викликається функція FinishVote, яка відповідає за закінчення голосування та підсумовування результатів. Слід зазначити, що це може зробити лише особа, яка подає питання на порядок денний.

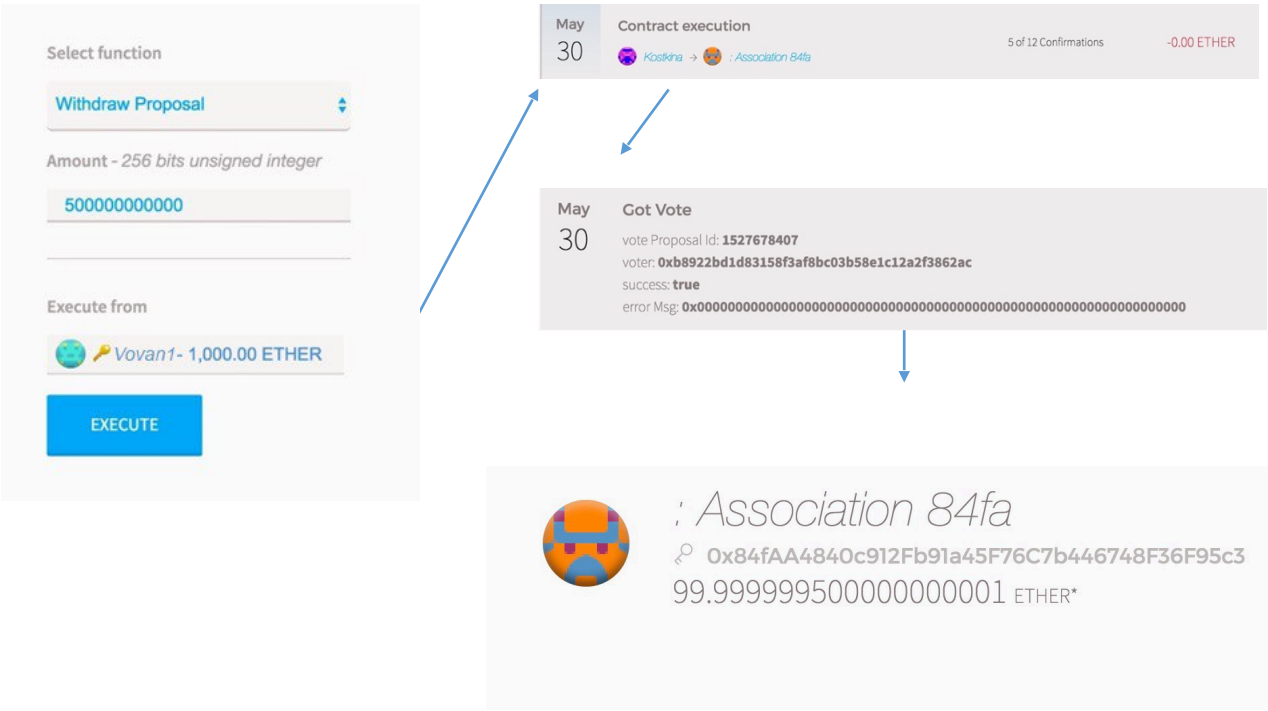

Рисунок 3.19 – Виконання функції переказу дивідендів

3.6 Робота веб-додатку

Пояснивши взаємодію архітектурних модулів та продемонструвавши, як це працює, нам необхідно описати розроблений веб-інтерфейс.

Розроблена програма є кросплатформеною і відповідає всім адаптивним принципам. Це означає, що ця програма може працювати на будь-якій машині та в будь-якій ОС.

Щоб протестувати ефективність, спробуємо створити смарт-контракт через веб-інтерфейс.

Початкове вікно програми показано на рисунку 3.20. Ми почнемо створювати контракт, натиснувши кнопку «Make a Contract».

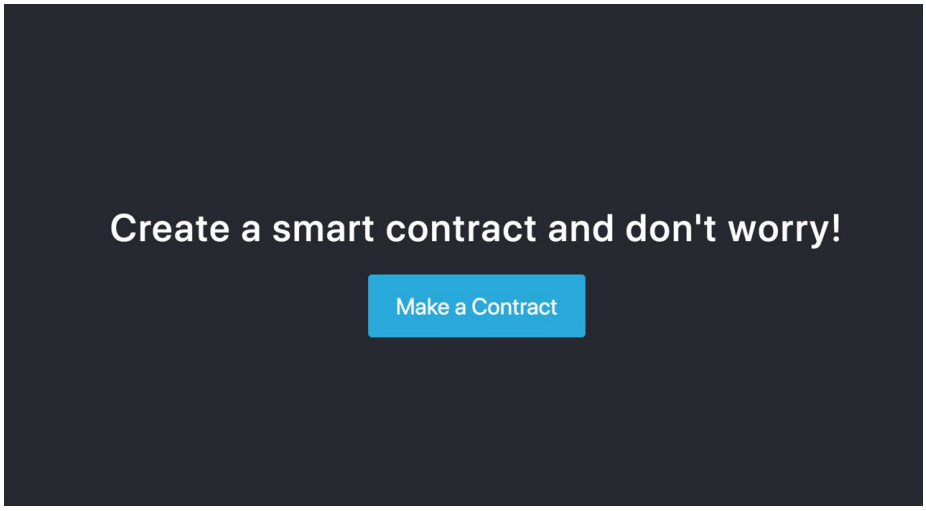

Рисунок 3.20 – Стартове вікно веб-додатку

У наступному вікні (Рис. 3.21) відображається форма введення інформації про акціонерів. Наприклад, ми створимо двох. Для цього потрібно внести дані у всі поля форми: імена, посади, доля та особисті адреси криптогаманця ETH (Рис. 3.21). Зазначу, що всі введені дані є абсолютно анонімні, тобто ніхто не може їх переглядати та використовувати в майбутньому. Натисніть на “Add new patrner", щоб зареєструвати іншого акціонера (Рис. 3.22). Якщо у підприємства є більше двох співвласників, ви можете змінити кількість областей вводу даних, натиснувши відповідну кнопку.

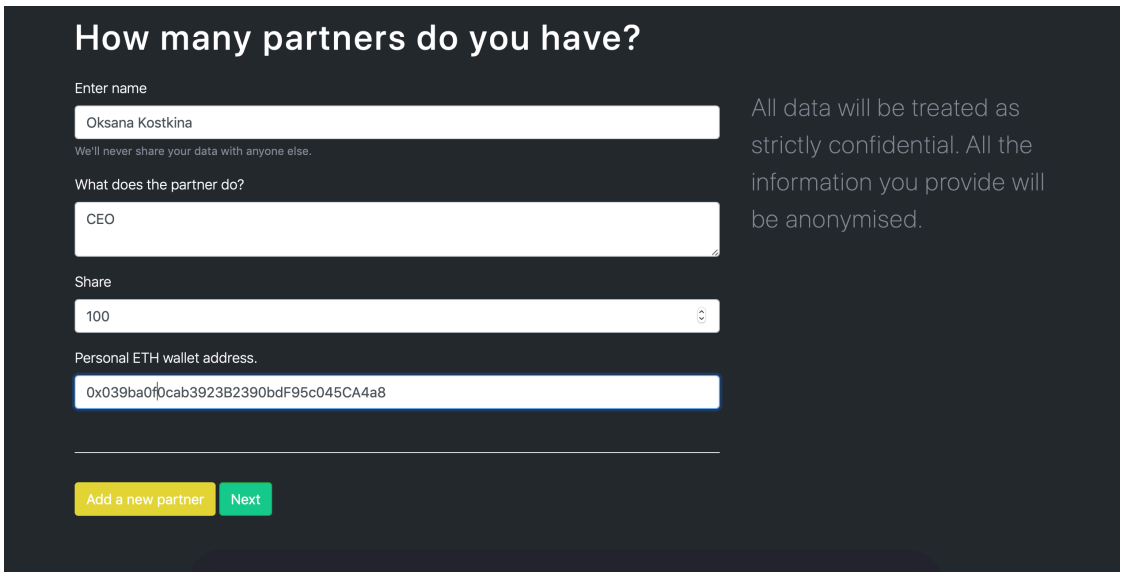

Рисунок 3.21 – Перший співвласник

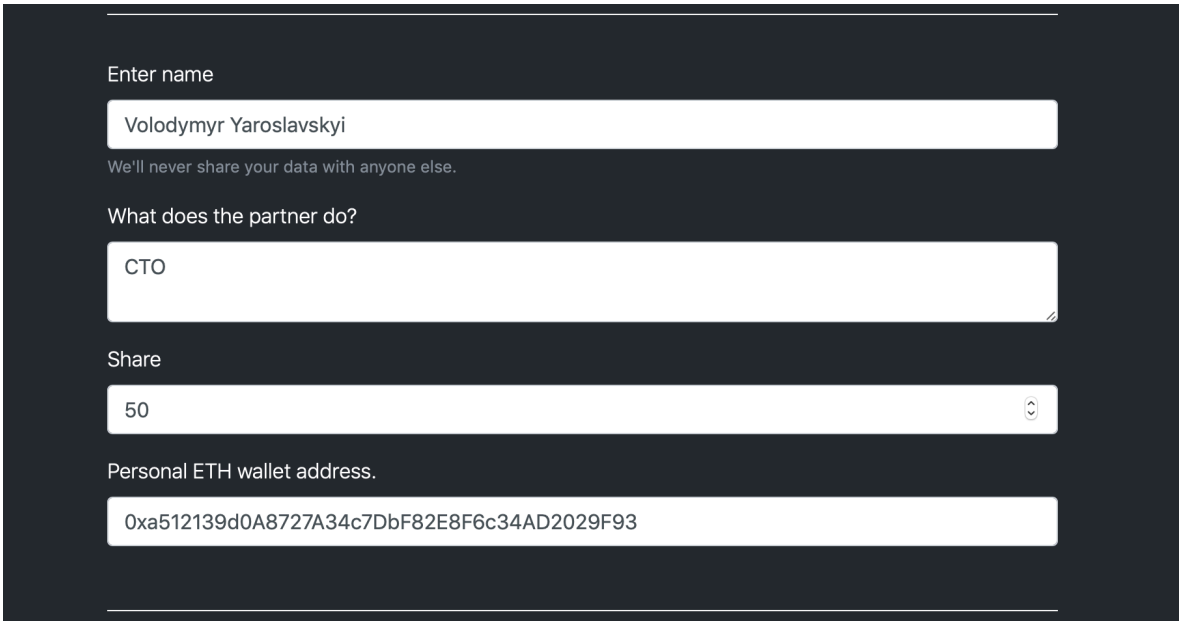

Рисунок 3.22 – Другий співвласник

Якщо користувач введе невірні дані, браузер видасть відповідне сповіщення. Якщо дані правильні, має з'явитися вікно для перевірки даних перед відправленням на блокчейн (Рис. 3.23). На цьому етапі ви все ще маєте можливість змінити дані про акціонерів якщо щось подано невірно.

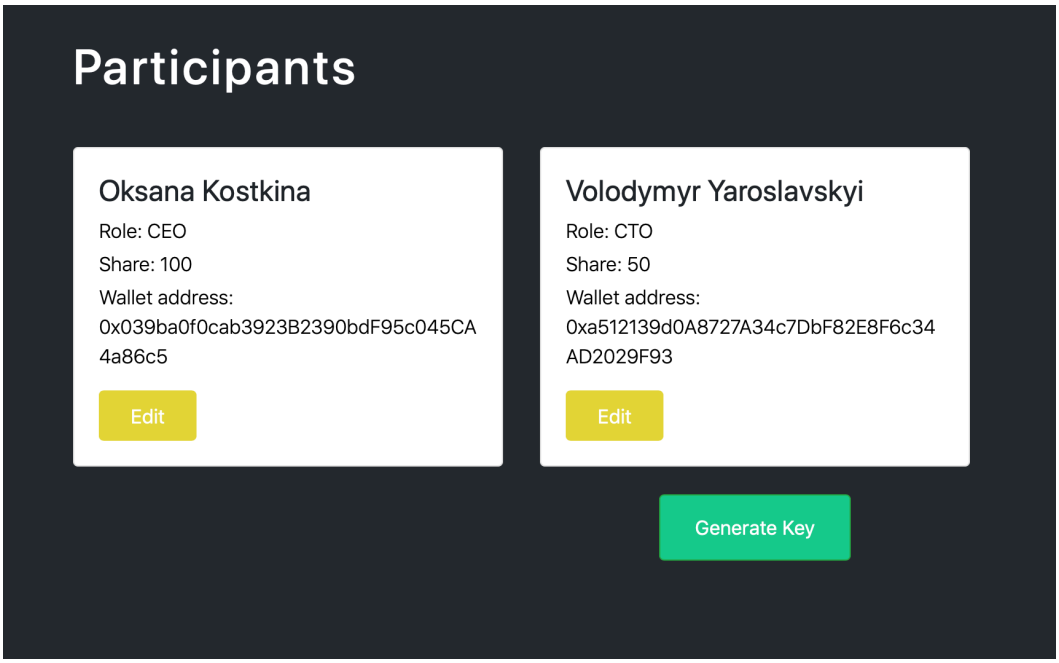

Рисунок 3.23 – Вікно з перевіркою інформації про співвласників

Після перевірки правильності інформації, введеної на даному етапі, натисніть «Generate Key», щоб створити смарт-контракт і занести його в блокчейн. Якщо все вдасться, з'явиться відповідне повідомлення (Рис. 3.24).

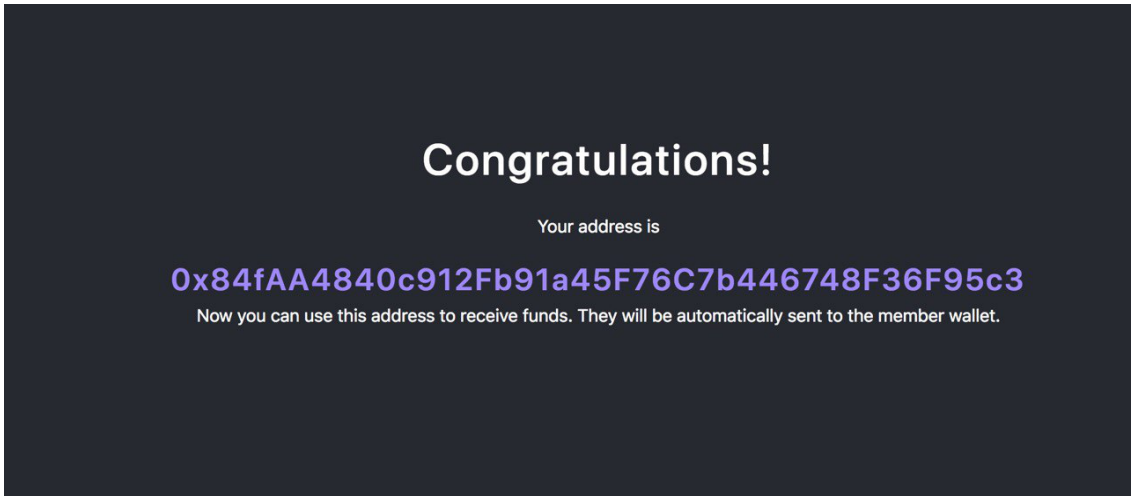

Рисунок 3.24 – Повідомлення про успішне створення смарт-контракту

Щоб перевірити контракт, давайте перейдемо до гаманця Mist, щоб дізнатись, чи існує контракт за цією адресою. У верхній частині Рисунка 3.25 ви можете побачити хеш-ключ, створений за допомогою розумного контракту, що відповідає попередньому зображенню (Рис. 3.24).

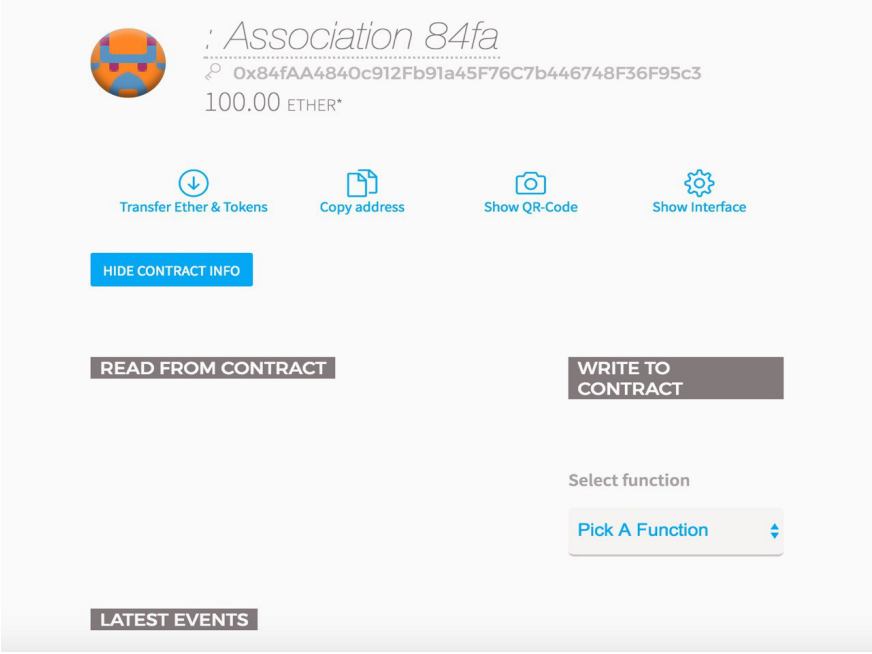

Рисунок 3.25 – Новостворений смарт-контракт

Натисніть “Contract Info”, щоб переглянути дані про співвласників компанії щойно створеного контракту. На Рис. 3.26 ви можете побачити shareholderNames на полі, яка описує долю розподілу серед акціонерів. У полі shareholderShares об'єкт з деталями дистрибутиву передається, і хеш-ключ контракту відображається в останньому полі. Всі хеші є такими самими, як і хеші, розміщені у формі, тому можемо стверджувати що програма працює нормально , а смарт-контракт створюється правильно.

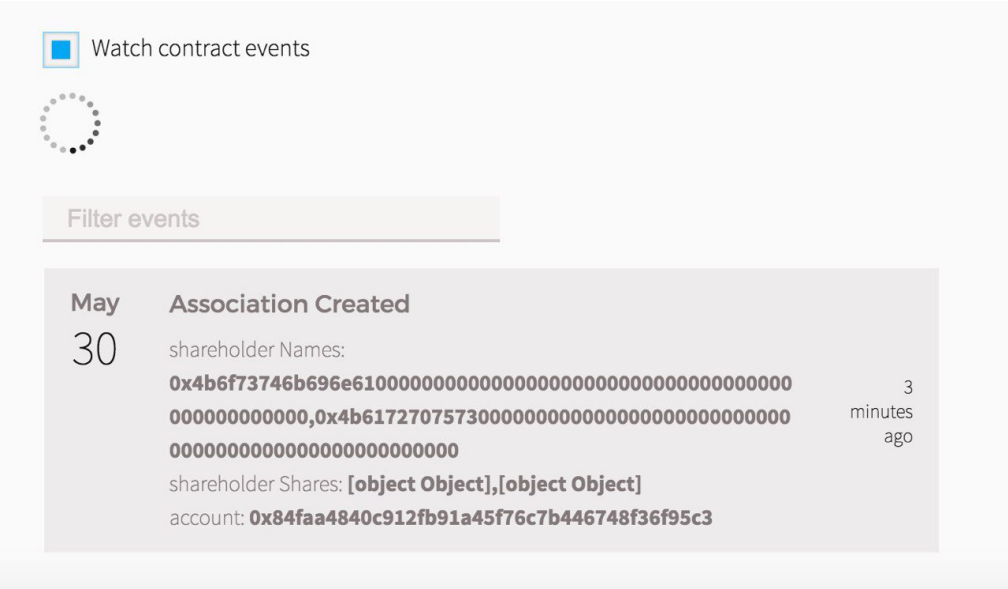

Рисунок 3.26 – Деталі контракту правильно

Після цих дій, деталі новоствореного смарт-контракту зберігаються в приватному blockchain Ethereum.

Висновки за розділом 3

В даному розділі було проаналізовано роботу смарт-контракту, який виконувався в мержі приватного блокчейну, а також роботу веб-інтерфейсу до нього.

Основним етапом аналізу розробленого програмного рішення має бути порівняльний аналіз з конкурентами та визначення основних показників роботи програми. З наведених в першому розділі потенційних конкурентів було вибрано BitShares та AIRA як такі, які найбільш подібні до запропонованого рішення.

Підтримання рішень задачі розподілення коштів:

- Даний проект: підтримує;
- BitShares: підтримує;
- AIRA: підтримує;

Підтримання рішень задачі перерозподілу акцій:

- Даний проект: підтримує;
- BitShares: підтримує з обмеженнями;
- AIRA: не підтримує;

Ціна впровадження:

- Даний проект: в межах 1-2 ETH;
- BitShares: Open Source, але для впровадження прийдеться набирати команду спеціалістів;
- AIRA: Open Source;

Варто зазначити, що так як смарт контракти працюють на розподілених реєстрах, то для економічного аналізу для них необхідно проводити також аналіз використання ресурсів обладнання, що виражається в абсолютних одиницях GAS. Ціна виконання задачі ініціалізації підприємства:

- Даний проект: ~ 21,000 gas;
- BitShares: неможливо порахувати, так як використовується власний блокчейн;
- AIRA: ~ 52,000 gas;

Тому, в довгостроковій перспективі ми бачимо що розроблений та запропонований нами проект твердо займає свою нішу з точки зору покриття підзадач та є економічно вигідний для впровадження у підприємство.

Також, в даному розділі був продемонстрований повний цикл роботи програми, тому даний розділ може бути основою до створення користувацького мануалу для майбутніх користувачів.

4 СТАРТАП-ПРОЕКТ

Стартап - широко розповсюдження в останні роки форма ведення підприємницької діяльності, що є популярною за рахунок зниження бар'єрів виходу на ринок (наприклад, Інтернет як інструмент продажів між кордонами, що дозволяє легко виходити на міжнародні ринки), вважається складовою інноваційної економіки. Ця концепція тотально зростає в якості за рахунок інноваційних інструментів мобільності, гнучкості та великого числа нових запусків стартап-проектів.

Створення та впровадження стартап-проектів на ринок знаменує високий рівень ризиків, успішними стають лиш лиш невелика частка, 10% - 20% . Сама по собі ідея прокту коштує майже нічого. На початковому етапі, важлива наявність особи, відповідальної за вирішення основних задач проекту.

Старт успішного стартапу залежить від виконання низки кроків:

а) Маркетингове дослідження стартап-проекту. На цьому етапі:

- розробити опис самої концепції проекту та визначити загальний напрямок використання потенційних продуктів або послуг та їх відмінності від конкурентів;
- проаналізувати ринкові можливості для її реалізації;
- на основі аналізу ринкового середовища розробити стратегію впровадження в ринок потенційних продуктів у рамках проекту.

б) Організація операційної діяльності стартап-проекту. На цьому етапі:

- підготовка плану впровадження для початку проекту;
- розрахувати попит на основні засоби та нематеріальні активи;
- визначити запланований обсяг виробництва потенційних продуктів, і на цій основі розвивати попит на матеріальні ресурси та персонал;

- обчислити загальну вартість запуску проекту та планові загальні економічні витрати, необхідні для реалізації проекту.

в) Фінансово-економічне дослідження та дослідження ризиків проекту. На цьому етапі:

- визначити суму інвестиційних запозичень;
- обчислити основні фінансово-економічні показники (дохідність, вартість виробництва, ціна реалізації, податкове навантаження та чистий прибуток) проекту та визначити індекс інвестиційної привабливості проекту (фінансовий резерв сили, прибутковість продажів та інвестицій, термін окупності проекту);
- визначити рівень ризику проекту та визначити основні ризики проекту та способи його запобігання (відповідь на ризик).

г) Комерціалізація проекту. Цей етап передбачає:

- визначити цільові групи інвесторів та описати їх бізнес-інтереси;
- скласти інвестиційну пропозицію (пропозиції): попередній опис проекту, щоб забезпечити початкове розуміння відносин між інвестором та проектом;
- план для сприяння заходам: визначити каналіи зв’язку, платформи та планувати рекламні системи на окремих каналах;
- планувати ресурси для здійснення заходів щодо впровадження.

Ці фази послідовно реалізуються та створюють передумови для успішного розвитку проекту. Проте в процесі створення та розвитку на початкових стадіях експерти відповідно відзначили, що відсутність маркетингових знань та навичок з ведення стартапу, а також щодо розвитку популярності ринкових проектів є основною причиною стартапів з високим рівнем банкрутства, і цю проблему можна вирішити шляхом навчання кофаундерів. Тому основною метою цих методів є надання студентам знань про маркетинг та економіку промислового сектору, основних принципів

стратегії маркетингового розвитку стартапів, використовуючи ефективні маркетингові інструменти для просування виробництва високотехнологічної продукції та послуг.

4.1 Ідея проекту

Аналіз, що проводиться в рамках цього підпункту представлений у формі таблиці:

- зміст пропонованої ідеї;
- напрямки застосування;
- основні переваги, які можуть отримати користувачі продукту (у кожному напрямком застосування);
- відмінності від існуючих аналогів та альтернатив,

Перші три етапи представлені в табличній формі (табл. 4.1), дають повне уявлення щодо змісту ідеї та можливого основного потенційного ринку, де необхідно знайти потенціал цільової аудиторії.

Таблиця 4.1 – Ідея проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Система керування коштами і дольовою участю в підприємствах	Надання компаніям можливісь провести розподіл коштів за дивідендами	Прозорість процесу та відсутність можливостей для махінацій, фальсифікацій та обману.
	Надання співвласникам можливості продавати/ купувати/ обмінювати акції	Повністю захищенні домовленості та відсутність можливості “розмивання”

Аналіз основних технічних та економічних переваг ідеї (в порівнянні існуючих аналогів та альтернатив) по відношенню до конкурентів включає:

- визначення переліку техніко-економічних характеристик та характеристик концепції;
- визначення попереднього діапазону конкурентів (конкурентних проектів) або альтернативних продуктів або товарних аналогів, які вже існують на ринку, а також збирання технічних та економічних показників для власної ідеї проекту відповідно до переліку визначених вище параметрів конкурентів;
- порівняльний аналіз показників: для власних концепцій показники визначаються як: a) гірші значення (W, слабкі); b) аналогічні (N, нейтральні) значення; в) оптимальні значення (S, сильні) (табл. 4.2)

Таблиця 4.2 – Визначення характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/ концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Власний проект	BitShares	AIRA			
1.	Ціна	Низька	Висока	Висока			+
2.	Ефективність	Висока	Висока	Висока		+	
3.	Функціонал	Вузький	Широкий	Широкий	+		

Визначення недоліків, сильних та нейтральних характеристик потенційних ідей продукту є основою аналізу їх конкурентоспроможності.

4.2 Технологічний аналіз проекту

В межах цього підрозділу було проведено технічний аудит, за допомогою якого можна було б реалізувати концепцію проекту (технологію створення продукту).

Визначення технічної доцільності концепції проекту передбачає аналіз наступних компонентів (табл. 4.3):

- Яка технологія проектування продуктів відповідно до концепції проекту?
- Чи є така технологія, яка ще потребує розвитку / вдосконалення?
- Чи доступні ці технології для авторів проекту?

Таблиця 4.3 – Технологічний аналіз проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1.	Розробка ПЗ, а смарт контракт, та веб-інтерфейсу до нього.	Solidity+Scala	Наявна	Доступна
2.		Serpent+Scala	Наявна	Недоступна
3.		Solidity+JavaScript	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: Solidity+Scala				

Згідно з аналізом, наведеним у таблиці 4.3, можна зробити висновок про можливість реалізації проекту. Технічно такі технології, як Scala + Solidity, доступні завдяки їх зручності та відкритому коду.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можуть бути використані протягом періоду реалізації проекту на ринку, а також ринкові загрози, які можуть перешкоджати реалізації проекту, дозволяють планувати розробку проекту з урахуванням стану ринкового середовища, потреб потенційних клієнтів та вад конкурентів.

Спочатку проводився аналіз попиту: попит, кількість, динаміка ринку (табл. 4.4).

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1.	Кількість головних гравців, од	0
2.	Загальний обсяг продаж, ВТС/ум.од	1
3.	Динаміка ринку (якісна оцінка)	Зростає
4.	Наявність обмежень для входу (вказати характер обмежень)	Висока конкуренція
5.	Специфічні вимоги до стандартизації та сертифікації	Немає
6.	Середня норма рентабельності в галузі (або по ринку), %	70%

Середня рентабельність галузі порівнянніюємо з ставкою депозиту банку. Значення більше, тому має сенс вкладати кошти в цей проект.

Згідно з аналізом, наведеним у таблиці 4.4, можна зробити висновок, що ринок є привабливим для виходу на нього.

Було сформовано потенційні групи клієнтів, їх характеристики та складено індикативний перелік вимог до кожної групи (табл. 4.6).

Таблиця 4.5 – Характеристика потенційних клієнтів стартап проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Потреба в автоматизованій системі розподілу коштів і дольових забовязань	Підприємці, які є членами акціонерних товариств	Необхідність невисокої ціни продукту	Вимоги до точності та ефективності роботи системи
2.				

Визначивши потенційну клієнтську базу, було проаналізовано ринкове середовище: сукупність факторів, які сприяють впровадженню ринкового проекту та факторів, що перешкоджають його розвитку (табл. 4.6, 4.7).

- ринкові можливості є середовищем, яким компанії можуть скористатися. Ринкові можливості можуть призвести до погіршення позицій конкурентів, різкого підвищення попиту, появи нових технологій та підвищення рівня доходів. Слід зазначити, що можливість SWOT-аналізу - це не всі можливості, які існують на ринку, а лише ті, які можуть бути використані.
- ринкові загрози - події, які можуть негативно вплинути на бізнес. Приклади ринкових загроз: виход на ринок нових конкурентів, збільшення податків, зміна смаків клієнтів, зниження рівню народжуваності тощо.

- коефіцієнти попиту (враховуючи спроможність ринку, швидкість зростання або зменшення, структура попиту на продукцію підприємства тощо);
- конкурентоспроможні чинники (слід враховувати кількість основних конкурентів, наявність альтернатив на ринку, висоту вхідних і вихідних бар'єрів, розподіл частки ринку серед основних учасників ринку тощо);
- фактори збуту (необхідно звернути увагу на кількість посередників, наявність розподільних мереж, умови постачання матеріалів і комплектуючих тощо);
- економічні фактори (обмінний курс гривні (долар США, євро), рівень інфляції, зміна рівня доходу, національна податкова політика тощо);
- політико-правові чинники (розрахункові рівні політичної стабільності в країні, рівень легальної грамотності населення, рівень відповідності, ступінь корупції влади тощо);
- наукові та технологічні чинники (як правило, враховуючи розвиток науки, ступінь впровадження інноваційного промислового виробництва (нових продуктів та технологій), держава підтримує науку і Т. та ін).
- соціально-демографічні чинники (звертайте увагу на розмір та стать, вікову структуру населення в регіоні, де працює компанія, рівень народжуваності та рівень смертності у сфері зайнятості);
- соціокультурні фактори (як правило, враховуючи традиції та системи соціальних цінностей, існуючу споживчу культуру товарів і послуг, наявні стереотипи поведінки людей тощо);
- природні та екологічні чинники (з урахуванням кліматичної зони господарської діяльності, умов навколишнього середовища, відносин між громадськістю та охороною навколишнього середовища тощо);

- міжнародні фактори (включаючи ступінь світової стабільності, існування локальних конфліктів тощо).

Таблиця 4.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Новий продукт	Потенційні користувачі з підозрою ставляться до нових продуктів	Поширення рекламної кампанії

Таблиця 4.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Потреба у недорогому продукті	Потреба компаній у недорогому та зрозумілому рішенні у сфері розподілення капіталу	Задоволення потреби компаній у недорогому та зрозумілому рішенні у сфері розподілення капіталу

Надалі було проведено дослідження пропозиції: визначено загальні характеристики конкурентної ситуації на ринку (таблиця 4.8).

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: олігополія	У сфері домінує невелика кількість компаній	Поширення рекламної кампанії
2. За рівнем конкурентної боротьби: міжнаціональний	Наявна міжнаціональна конкуренція	Поширення рекламної кампанії
3. За галузевою ознакою: внутрішньогалузева	Наявна конкуренція в рамках одної галузі	Можливість вийти на ринок з недорогим продуктом
4. Конкуренція за видами товарів: товарно-видова	Наявна конкуренція між схожими продуктами	Можливість вийти на ринок з недорогим продуктом
5. За характером конкурентних переваг: нецінова	Наявна конкуренція завдяки підвищенню якості та надійності продукції	Можливість вийти на ринок з недорогим продуктом
6. За інтенсивністю: не марочна	Наявна конкуренція, де роль торгової марки незначна	-

Було проведено аналіз конкуренції у галузі за моделлю М. Портера (таблиця 4.9). М. Портер вирізняє п’ять основних факторів, що впливають на привабливість вибору ринку з огляду на характер конкуренції. Це (Рис 4.1):

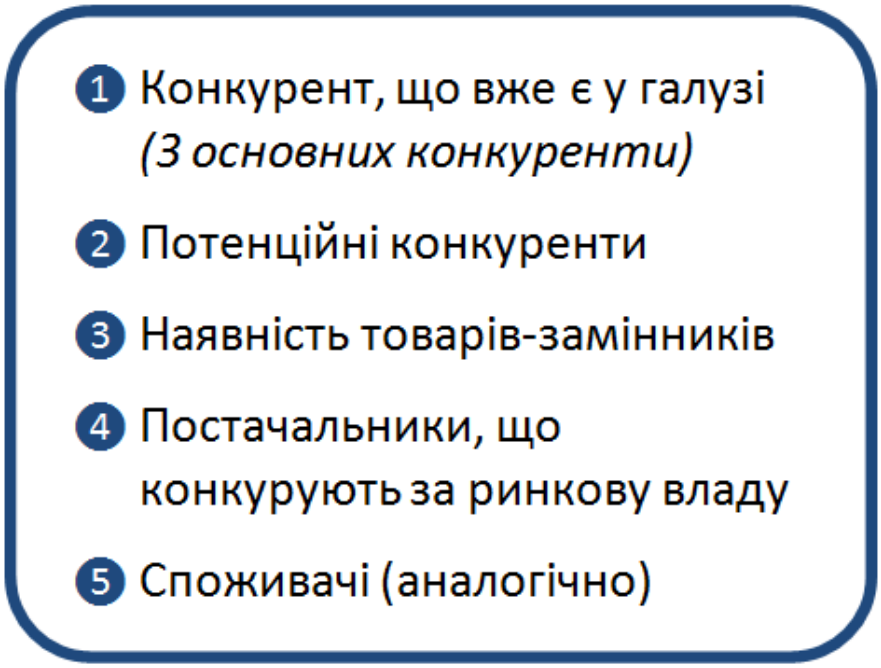

Рисунок 4.1 – Модель Портера

Сильна позиція кожного фактора вказує на її здатність забезпечити необхідний оборот капіталу та її здатність впливати на агентів на інших ринках та контролювати співпрацю. Характеристики модельних факторів різних галузей різні і залежать від часу. Сила кожного фактора полягає в структурі галузі, впливу на неї технічних та економічних характеристик.

Згідно з результатами аналізу в таблиці 4.9, можна зробити наступні висновки.

Враховуючи конкурентну ситуацію, є можливість працювати на ринку. В наступному параграфі розкритий перелік факторів конкуренції.

На основі аналізу конкурентів, таблиця 4.9, а також характеристик проекту (див. Табл. 4.2) та факторів ринкового середовища (табл. 4.6, 4.7) отримуємо список конкурентних факторів. Аналіз проводився згідно таблиці 4.10.

Таблиця 4.10 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	BitShares AIRA	Доступ до каналів розподілу	-	Розмір закупівель, контроль якості	Ціна
Висновки	Висока інтенсивність конкурентної боротьби з боку прямих конкурентів	Є можливості входу в ринок. Потенційних конкурентів немає.	Постачальники не диктують умови роботи на ринку.	Клієнти диктують високі критерії якості продуктів.	-

За визначеними факторами конкурентоспроможності (таблиця 4.11) проведено аналіз сильних та слабких сторін стартап-проекту (таблиця 4.12).

Таблиця 4.11 – Обгрунтування факторів конкурентноспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1.	Ціна	Ціна запропонованого продукту значно нижче ніж ціни конкуруючих, це значно поширює потенційну клієнтську базу
2.	Ефективність	Результати використання продукту в умовах стохастичної невизначеності є сумірними або кращими за результати конкуруючих продуктів
3.	Поріг входження	Так як у сфері тип конкуренції є міжнаціональний, достатньо складно вивести новий невідомий невеликий продукт на ринок

Таблиця 4.12 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим рішенням						
			–3	–2	–1	0	+1	+2	+3
1.	Ціна	15	+						
2.	Ефективність	18			+				
3.	Поріг входження	10							+

Останній етап аналізу ринку можливостей реалізації проекту - підготовка SWOT-аналізу (матриця аналізу міцності та вразливості)

Список ринкових загроз та ринкових можливостей складається шляхом аналізу загроз та факторів можливостей маркетингового середовища. Наприклад, зниження доходу потенційних покупців - фактор загрози, який збільшить важливість цінового фактора при виборі товару і відповідно цінової конкуренції (це - загроза ринку).

Таблиця 4.13 – SWOT аналіз стартап проекту

Сильні сторони: Ціна продукту Ефективність продукту	Слабкі сторони: Невідомість продукту
Можливості: Охоплення аудиторії, що не може дозволити дорогі інтелектуальні комплекси	Загрози: Можлива незацікавленість продуктом через його невідомість та невеликість

На підставі SWOT-аналізу були розроблені альтернативні варіанти ринкової діяльності (перелік заходів) для ринкової оптимізації часу запуску

проекту на ринок в залежності від потенційних проектів конкурентів, які були розміщені на ринку (див. Таблицю 4.9, Потенційні конкуренти).

Альтернативи визначаються на основі умов та аналізу імовірності отриманих ресурсів (таблиця 4.14).

Таблиця 4.14 – Альтернативи ринкового впровадження стартап проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Розробка програмного забезпечення та грамотна маркетингова програма	Велика	3-5 місяців

Після аналізу було обрано альтернативу No1.

4.4 Розроблення ринкової стратегії проекту

Першим кроком у розробці ринкової стратегії є визначення стратегії охоплення ринку: опис цільової групи потенційних клієнтів (табл. 4.15).

Таблиця 4.15 – Вибір цільових груп потенційних клієнтів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Фізичні особи та невеликі компанії, зацікавлені в інвестуванні	Значна готовність	Високий	Низька	Середня
2.	Великі компанії, зацікавлені в інвестуванні	Незначна готовність	Низький	Висока	Висока
Які цільові групи обрано: Фізичні особи та невеликі компанії, зацікавлені в інвестуванні					

Аналіз потенційних груп споживачів вибраної цільової аудиторії, яку цей продукт забезпечує, і стратегії охоплення ринку - централізована маркетингова стратегія (компанія зосереджується на одному сегменті).

Для роботи в певному сегменті ринку було розроблено базову стратегію розвитку (табл. 4.16).

Згідно М. Портера, існують три основні стратегії розвитку, які характеризуються охопленням цільового ринку та тими конкурентними перевагами, які мають бути досягнуті на ринку (вартість або якість продукції).

Стратегія лідерства на витратах показує, що компанія за рахунок внутрішнього та / або зовнішнього середовища може забезпечити більший запас маржинальності між вартістю товару та середньою ринковою ціною (або ціною її основного конкурента). Зокрема, ця стратегія передбачає, що за

рахунок великих обсягів компанія може досягти скорочення витрат через великі можливості оптимізації. Ця стратегія часто тісно пов'язана з ростом і масштабуванням.

Компанії, які вибирають цю стратегію, будуть уважно стежити за постійними витратами, скорочувати витрати на виробництво, продаж та рекламу, інвестувати в скорочення витрат та ретельно проектувати нові продукти.

Переваги стратегії:

- навіть у випадку цінової війни компанія може протистояти своїм прямим конкурентам і матиме можливість отримувати прибуток з нижчою ціною ніж у своїх конкурентів;
- сильні клієнти не можуть знижувати ціни нижче прийнятних рівнів для найсильніших конкурентів;
- низькі витрати захищають від потужних постачальників, оскільки вони пропонують велику гнучкість для компанії при підвищенні витрат;
- низька вартість створює перешкоди для входження початківців, а також забезпечує хороший захист альтернатив.

В конкурентній боротьбі, використовуючи цю стратегію, менш ефективні компанії будуть змушені залишити ринок

Стратегії спеціалізації передбачають зосередження уваги на потребах цільового сегмента ринку, а не намагання охопити весь ринок. Мета полягає в тому, щоб краще відповідати потребам вибраних цільових сегментів, ніж конкуренти. Ця стратегія може ґрунтуватися на перевазі диференціації та витрат, з одного боку чи іншого, але тільки в цільовому сегменті ринку. Проте, якщо стратегія не реалізується, а частка на ринку є низькою, це серйозно вплине на конкурентоспроможність компанії.

Таблиця 4.16 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1.	Розробка програмного забезпечення та грамотна маркетингова програма	За рахунок потреби в дешевому продукті	Ціна	Стратегія спеціалізації

Наступним кроком є стратегія конкурентної поведінки (табл. 4.17).

Залежно від ступеня формування ринку, конкуренція провідних компаній власності вибрати один з трьох стратегій: розширення первинного попиту, оборонних або наступальних стратегій або застосувань демаркетинг та диверсифікації.

Стратегія розширення первинного попиту є обґрунтованою, якщо фірма-лідер витрачається на боротьбу проти невеликих конкурентів, це може суттєво вплинути на економічну експансію на початкових етапах попиту. У цьому випадку компанія вживає заходів для формування (споживчі товари, що використовуються у експансії, формують постійні потреби, збільшують одноразове споживання), а також сприяння новим сферам застосування існуючих продуктів та відкриття нових клієнтських груп. Ця стратегія, швидше за все, буде лише на початку життєвого циклу, коли попит більше масштабований і конкурує з меншим тиском. Інакше лідери компанії повинні прийняти оборонні або агресивні стратегії.

Завдяки своїй а інноваційній позиції на ринку, лідери почали бути атакованими імітаторами. У такому випадку компанія може обрати оборонну стратегію, мета якої - захистити свою частку ринку. Захист може бути:

- створення технологічних бар'єрів для інновацій,;
- розширення асортименту;
- Проведення цінових війн та / або проведення масштабних рекламних атак.

Наступальна стратегія передбачає збільшення своєї частки на ринку. Ця мета полягає у подальшому вдосконаленні рентабельності компанії на ринку шляхом максимального використання економії на масштабі. Однак існує межа для подальшого зростання частки ринку та отримання прибутку. Наступальна стратегія передбачає агресивну інноваційну політику компанії. Вона продовжує атакувати поки розрив між ними та їх основними конкурентами не зросте достатньо. Постійне технічне та економічне вдосконалення, зміна розміру та форми упаковки, використання подій, маркетинг - типовий арсенал компонентів бізнес-лідерів.

Якщо фірма потрапляє під дію антимонопольного законодавства, вона може удатися до стратегії демаркетинга, що припускає скорочення своєї частки ринку, зниження рівня попиту на деяких сегментах за рахунок підвищення ціни. При цьому ставиться завдання недопущення на ці сегменти конкурентів, а компенсація втрат прибутку через зменшення обсягів виробництва компенсується встановленням вищих цін.

Проте у більшості випадків найпривабливішою стратегією для компаній-лідерів є диверсифікація, що дозволяє використати переваги масштабу виробництва, know – how.

Стратегія конкурентної ніші. При прийнятті стратегії зайняття конкурентної ніші (інші назви - стратегічні фахівці або нішери) компанії вибирають один або декілька ринкових сегментів. Головною особливістю є

невеликий розмір сегментів. Ця конкурентоспроможна стратегія походить від базових стратегій, таких як концентрація. Ніша, щоб зробити її привабливою для підприємств, повинна відповідати наступним умовам (табл. 4.17.):

- забезпечує достатньо прибутку, щоб зробити доцільні виробничі та сервісні процеси;
- тримайється стабільно протягом тривалого часу;
- має бути добре захищена і мати високий поріг входу;
- не приваблива для конкурентів;
- підходить за з цілями та ресурсами компанії, а також з його конкретними можливостями.

Головне завдання для обраної стратегії нішера чи професійної компанії - це його подальша спрямованість на конкурентну перевагу, лояльність та прихильність до підтримки клієнтів та розвиток бар'єрів входу ринку.

Таблиця 4.17 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1.	Так	Шукати нових споживачів	Буде розроблений продукт зі схожим функціоналом, але іншою методологією	Стратегія заняття конкурентної ніші.

Позицювання — це маркетингове забезпечення товарові бажаного місця на ринку і у свідомості потенційних покупців (образ). Позиція компанії чи продукту показує чим він унікальний УТП (унікальну торговельну пропозицію), чим відрізняється від конкурентів (відстройка від конкурентів), чим корисний споживачу. При продажі продукту це поєднання матеріальних (розмір, колір, вага, швидкість і т. п.) І нематеріальних (престиж, мода, подарунок і т. п.) атрибутів.

Дослідження показали, що якщо позиціонування зазначених вище трьох критеріїв виконується, це неефективно, оскільки у свідомості споживачів немає осадження.

Таблиця 4.18 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1.	Необхідність невисокої ціни продукту, точності та ефективності роботи СППР	Стратегія спеціалізації	Ціна та ефективність	Низька ціна Висока ефективність Простота у використанні

Результатом виконання підрозділу стала узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначає напрями роботи стартап-компанії на ринку.

4.5 Розроблення маркетингової програми стартап-проекту

Задача сформувати маркетингову концепцію продукту, який отримає споживач. З цією метою в таблиці 4.19 підсумовані результати попереднього аналізу конкурентоспроможності продукції.

Концепція продукту - це характеристика фізичних та інших характеристик товарів, а також ряд переваг, які він обіцяє певним групам споживачів.

Таблиця 4.19 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1.	Потреба в недорогому продукті	Пропонує недорогий продукт	Ціна товару нижче ніж ціна товару конкурентів
2.	Потреба в ефективному продукті	Пропонує ефективний продукт	Ефективність товару вище ніж ефективність товару конкурентів

Розроблена трирівнева модель маркетингового продукту: концепції продукту та / або послуги, її фізичні компоненти та, особливо, процес підготовки (див. Табл. 4.20).

Рівень 1:

Постановка задачі, яку вирішує продукт.

Рівень 2:

Як цей рівень продукту реалізується в реальних рішеннях / включає в себе якісні характеристики, дизайн, упаковку та ціну.

Рівень 3:

Підтримка товару (підтримка) - Додаткові послуги та переваги, що базуються на дизайні продукту, а споживачі продукції - додаткову цінність (забезпечення якості, доставка, умови оплати та інше).

Таблиця 4.20 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Недорога система розподілу коштів та акцій підприємства</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/ Е/Ор</td></tr>
<tr><td>1. Мультиплатформенність
2. Зручний інтуітивний інтеріейс</td><td>-</td><td>-</td></tr>
<tr><td colspan="3">Якість: стандарти ефектиіності</td></tr>
<tr><td colspan="3">Пакування: електронне розповсюдження</td></tr>
<tr><td colspan="3">Марка: Blockchain handshake</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу: -</td></tr>
<tr><td colspan="3">Після продажу: технічна підтримка</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: криптографічний механізм захисту сирцевого коду</td></tr>
</table>

Після створення маркетингової моделі для вашого продукту, слід передбачити, що за допомогою ноу-хау проект не буде скопійований.

Наступним кроком є визначити межу ціни, якою слід керуватись при встановленні ціни потенційного продукту (остаточне визначення ціни має місце під час фінансово-економічного аналізу проекту), включаючи аналіз ціни аналогічних товарів, а також аналіз цільової групи споживачів. Рівень доходів (табл. 4.21). Цей аналіз проводився експертними методами.

Таблиця 4.21 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	-	7500000-54000000 грн.	25000+ грн.	8000-18000 грн.

Наступним кроком є оптимізація системи збуту, в межах якого буде прийняте рішення (таблиця 4.22):

- збут власноруч чи залучення сторонніх посередників.
- користуватися однорівневим каналом збуту;

Таблиця 4.22 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1.	Замовлення послуг через інтернет	Підтримка нормального функіонування сайту та ноди блокчейну	0	Електронне розповсюдження

Останньою складовою є розроблення концепції комунікацій, що основується на платформі для позиціонування, визначену специфіку поведінки клієнтів (таблиця 4.23).

Таблиця 4.23 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1.	Потреба в недорогому та ефективному продукті	Інтернет-мережі	Низька ціна Висока ефективність Простота у використанні	Провести якісну маркетингову кампанію	Донести специфіку продукту

Результатом роботи відділу є маркетингова програма, яка включає попередній аналіз концепцій продукту, продаж, просування та ціноутворення, виходячи з вартості та потреб потенційних клієнтів, конкурентної переваги ідей, стану та динаміки ринкового середовища, в якому реалізується проект, і належний вибір альтернатив ринкової поведінки.

Висновки за розділом 4

В даному розділі було проведено аналіз програмного продукту у якості стартап проекту. Можна зазначити що у проекта є можливість комерціалізації, адже ринок технологій віртуальної реальності динамічно розвивається, створюються нові додатки які, в свою чергу, стимулють попит на різноманітні контролери для управління контентом у віртуальній реальності.

На ринку наявна міжрегіональна конкуренція, існує багато компаній-конкурентів, тому вихід на нього не буде легким. Проте проект є доволі конкурентноспроможним завдяки, в першу чергу, своїй низькій собівартості. Через те, що він є повністю програмним, його розробка не потребує витрат на різноманітні матеріали та обладнання, необхідні для виготовлення корпусу,

схем, тощо.

Для впровадження ринкової реалізації проекту слід обрати альтернативу,

яка передбачає розробку програмного продукту, а потім якісну рекламу та PR, сконцентровану навколо позитивних характеристиках даного програмного продукту, таких як низька ціна, більш істотний ефект занурення, кросплатформеність, екологічність і т.д.

З огляду на проведений аналіз, можна чітко сказати, що подальша імплементація проекту є доцільною, адже він може знайти свою цільову аудиторію та зайняти місце на ринку.

ВИСНОВОК

Бурхливий розвиток стартапів та криптовалют дозволяє викорінити проблеми, що хвилюють підприємців з самого початку людської цивілізації. В даній роботі було запропоновано рішення для розподілу коштів підприємства та управління дольовою участю співвласників.

а) Перший розділ дослідження показує, що питання управління корпоративними фондами є як ніколи нагальним в наш час. Вдале керування фондами компаній є запорукою стабільного виконання забов’язань. Без планування та своєчасного аудиту транзакцій, підприємства неминуче зіштовхуються з проблемами, які можуть призвести до поступового бакнкрутства бізнесу.

б) Досліджено теоретичні основи різних систем керування грошовими коштами та вивчено вплив різних факторів на вирішення проблем управління активами. Тому використання технологій криптографії створило інноваційний підхід до вирішення цієї проблеми, так як дослідження показали, що технологія блокчейн має всі потрібні характеристики для того, щоб стати підгрунтям до створення новітньої системи розподілу коштів підприємства та управління дольовою участю співвласників.

в) Проаналізоване використання технології Blockchain в промисловості та розглянуто основні проблеми впровадження даної технології в підприємства, а саме:

- поганий показник проникнення криптовалют в підприємства;
- низький рівень загалбної криптотехнологічної грамотності та вмінь;
- недовіра до смарт-контрактів, написаних сторонніми компаніями.

При аналізі запропонованого рішення, було виявлено, що воно в тій чи іншій мірі усуває дані проблеми, а саме:

- реалізує простий інтерфейс для зручного користування людям з різним ступенем технічної підкованності;
- реалізує можливість розвертання на приватному блокчейні, що забезпечує безпечне середовище для тестування і роботи продукту.

Нажаль, показник проникнення криптовалют в підприємства дана робота виправити не може, але подібні проекти підігрівають інтерес бізнесу до блокчейну, тому дана система вносить свій вклад в цю справу.

г) У другому розділі досліджуються теоретичні основи, необхідні для вирішення проблеми. Тобто - докладний опис технології Blockchain, типи ланцюгів, технічні переваги та недоліки, взаємозв'язок монет і Blockchain. Після порівняння централізованих та розподілених баз даних, у другому випадку виявлено значні переваги.

ґ) Розглянуто концепцію смарт-контракту, проаналізовані платформи, що їх підтримують та вибрано платформу Ethereum. На сьогоднішній день, Ethereum є найбільш простим і розвиненим ланцюгом, та забезпечує розробників з широким спектром можливостей. Ethereum дозволяє розробляти смарт-контракти на мові програмування Solidity.

д) Проаналізовано практичну цінність смарт-контрактів, і результати підтверджують актуальність цієї технології на сьогоднішній день. Були також прийняті до уваги недоліки розумних контрактів, найважливішим з яких є відсутність юридичного статусу в більшості країн. У майбутньому це може стримувати швидкий технологічний розвиток технологій.

е) Третій розділ демонструє розробку розумних контрактів “from scratch”. Демонстрація показує, що завдання виконано, смарт-контракт та веб-інтерфейс до нього працюють як і очікувалось на етапі проектування. Смарт-контракти можна створювати та виконувати за допомогою розробленого веб-інтерфейсу та гаманця Mist, що робить його кросплатформеним та незалежним від будь-якого стороннього ПЗ.

є) Проведено аналіз роботи, замір основних показників працездатності та порівняння розробленого продукту з конкурентами, за результатами якого ми можемо стверджувати про його практичну цінність і новизну. Було виявлено, що продукт має потенціал до розвитку а також може бути змінений до вимог конкретного замовника.

ж) Було проаналізовано перспективи даного продукту в якості стартап-проекту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Blockchain technology – [Електронний ресурс]. – Режим доступу: https://blockgeeks.com/guides/what-is-blockchain-technology/.
2. Блокчейн – [Електронний ресурс]. – Режим доступу: https://ru.bitcoinwiki.org/wiki/Блокчейн.
3. Смарт-контракты: функции и применение – [Електронний ресурс]. – Режим доступу: https://cyberleninka.ru/article/n/smart-kontrakty-funktsii-i-primenenie.
4. A brief history in the evolution of blockchain technology platforms – [Електронний ресурс]. – Режим доступу: https://hackernoon.com/a-brief-history-in-the-evolution-of-blockchain-technology-platforms-1bb2bad8960a.
5. 55+ Corporations Working On Blockchain And Distributed Ledgers – [Електронний ресурс]. – Режим доступу: https://www.cbinsights.com/research/organizations-corporates-test-blockchains-distributed-ledgers/.
6. Биткоин-сервис Bitwage добавляет 18 новых валют – [Електронний ресурс]. – Режим доступу: https://bloomchain.ru/newsfeed/zarplatnyj-bitkoin-servis-bitwage-dobavlyaet-18-novyh-valyut/.
7. Bitshares – [Електронний ресурс]. – Режим доступу: https://coinmarketcap.com/ru/currencies/bitshares/.
8. Otonomos – [Електронний ресурс]. – Режим доступу: https://www.crunchbase.com/organization/otonomos.
9. Aira – [Електронний ресурс]. – Режим доступу: https://libre.life/dao/.
10. What is Blockchain – [Електронний ресурс]. – Режим доступу: https://www.ibm.com/blockchain/what-is-blockchain.
11. What is Blockchain technology – [Електронний ресурс]. – Режим доступу: https://www.cbinsights.com/research/what-is-blockchain-technology/.
12. Блокчейн технологія і платформа – [Електронний ресурс]. – Режим доступу:

https://uacrypto.top/blog/blockchain-guide.

13. Хеш-функція – [Електронний ресурс]. – Режим доступу: https://sites.google.com/site/anisimovkhv/learning/kripto/lecture/tema9
14. Public and private blockchain technology –[Електронний ресурс]. – Режим доступу: https://www.intheblack.com/articles/2018/09/05/private-public-blockchain.
15. What is blockchain and what can businesses benefit from it– [Електронний ресурс]. – Режим доступу: https://www.forbes.com/sites/forbesagencycouncil/2018/04/05/what-is-blockchain-and-what-can-businesses-benefit-from-it/.
16. Leverington A. Ethereum [Текст] / A. Leverington. – Portland: Rick Publishing, 2016. – 63 p.
17. Smart contracts – [Електронний ресурс]. – Режим доступу: https://www.coursera.org/learn/smarter-contracts.
18. Ethereum– [Електронний ресурс]. – Режим доступу: https://cointelegraph.com/tags/ethereum.
19. Introduction to Blockchain, Ethereum and Smart Contracts — Chapter 1 – [Електронний ресурс]. – Режим доступу: https://medium.com/coinmonks/https-medium-com-ritesh-modi-solidity-chapter1-63dfaff08a11/.
20. How to build an Ethereum Smart Contract for a Blockchain Marketplace – [Електронний ресурс]. – Режим доступу: https://rubygarage.org/blog/ethereum-smart-contract-tutorial.
21. ERC20 Token Standard – [Електронний ресурс]. – Режим доступу: https://theethereum.wiki/w/index.php/ERC20_Token_Standard/ .
22. Что такое токен стандарта Ethereum ERC20 – [Електронний ресурс]. – Режим доступу: https://bitfin.info/2110-standart-erc-20/.
23. Сравнение платформ для смарт-контрактов – [Електронний ресурс]. – Режим доступу: https://decenter.org/ru/sravnenie-platform-dlya-smart-kontraktov/.

24. Що таке Смарт-Контракти – [Електронний ресурс]. – Режим доступу: https://cryptobook.pro/shcho-take-smart-kontrakty.html.
25. Proof of Stake, Proof of Work – [Електронний ресурс]. – Режим доступу: https://www.bitdegree.org/tutorials/proof-of-work-vs-proof-of-stake/.
26. Proof of Work vs Proof of Stake – [Електронний ресурс]. – Режим доступу: https://www.sitepoint.com/proof-of-stake-vs-proof-of-work/.
27. Solidity – [Електронний ресурс]. – Режим доступу: https://solidity.readthedocs.io/en/v0.5.1/.
28. Solidity Smart Contracts: Build Dapps In Ethereum Blockchain – [Електронний ресурс]. – Режим доступу: https://www.udemy.com/solidity-smart-contracts-build-dapps-in-ethereum-blockchain/.
29. Погружение в разработку на Ethereum. Часть 2: Web3.js и газ [Електронний ресурс] – Режим доступу: https://habr.com/post/336770/.
30. Нода – [Електронний ресурс]. – Режим доступу: https://ru.bitcoinwiki.org/wiki/Нода.

ДОДАТОК А - ЛІСТИНГ ПРОГРАМИ

```
pragma solidity ^0.4.16;


contract Association {

  // Структура "Акціонер", що містить в собі повну інформацію щодо акціноера

  struct Shareholder {
    bytes32 name; // Повне ім'я акціонера
    bytes32 description; //Додаткова інформація - посада, паспортні дані тощо.
    address account; // Адреса персонального гаманця.
    uint share; // Цілочислений тип - кількість акцій акціонера
  }


  Shareholder[] shareholders; //Масив всіх акціонерів

  // Івент, що емітується при ініціалізації смарт-контракту
  event AssociationCreated(
    bytes32[] shareholderNames, // Імена всіх співвласників
    uint[] shareholderShares, // Кількість акцій всіх співвласників
    address account // адреса рахунку підприємства
  );

  //Івент, що емітується після того, як користувач проголосував за пропозицію
  event GotVote(
    uint voteProposalId, //Ідентифікатор пропозиції
    address voter, // Адреса користувача
    bool success, // Голос відданий успішно/не успішно
    bytes32 errorMsg // Повідомлення про помилку
  );

  //Івент, що емітується після того, голосування завершено
  event VotingResult(
    uint voteProposalId, //Ідентифікатор пропозиції
    bool success // Рішення прийнято/не прийнято
  );

  //Івент, що емітується після того, як був додоаний новий акціонер
  event ShareholderCreated(
    bytes32 shareholderName, // Повне ім'я акціонера
    uint shareholderShare, // кількість акцій акціонера
    address account // адреса персоанльного рахунку акціонаера
  );

  //Івент, що емітується після того, як були виплачені дивіденди
  event WithdrawFinished(
    bool success, // Розподілення дивідендів успішно/не успішно
    uint amount // Загальна сума дивідендів
  );

  //Івент, що емітується після того, як акції були переказані
  //(Перейшли від одного акціонера до іншого)
  event SharesTransfered(
    address from, // Адреса того, хто передав
    address to, // Адреса того, кому передав
    uint amount, // Загальна сума переказаних акцій
    bool success, // Переказ успішно/не успішно
    bytes32 errorMsg // Повідомлення про помилку
  );

    //Івент, що емітується після того, як кошти з корпоративного рахунку були
переказані
```

```
        event TransferResult(
          address to, // Адреса того, кому переказано
          uint amount, // Загальна сума переказаних коштів
          bool success, // Переказ успішно/не успішно
          bytes32 errorMsg // Повідомлення про помилку
        );

        event NewVoting(
          uint id,
          bytes description
        );

        //Івент, що емітується перед тим як контракт самоліквідується
        event ContractDestroyed();


        //Функція отримання інформації про поточну загальну кількість акцій
        function getTotalShares() private view returns (uint totalShares){
          totalShares = 0;
          for (uint i = 0; i < shareholders.length; i++) {
            totalShares += shareholders[i].share;
          }
          return totalShares;
        }

        // Службові масиви
        bytes32[] shareholderNames;
        uint[] shareholderShares;


        // Конструктор, який виконується при початковій ініціалізації контракту
        function Association() public payable {
          // Згенерований код ініціалізації початкових акціонерів
                  shareholders.push(Shareholder("Kostkina",  "CEO",
0xB8922bd1d83158F3AF8bC03B58e1C12A2f3862Ac, 100));
                    shareholders.push(Shareholder("Karpus",  "CTO",
0xa872C582e5b48Ec313286f1F06BA071Cc54300B7, 50));

          // Збір інформації та емітування івенту про створення
          for (uint i = 0; i < shareholders.length; i++) {
            shareholderNames.push(shareholders[i].name);
            shareholderShares.push(shareholders[i].share);
          }

          emit AssociationCreated(shareholderNames, shareholderShares, address(this));
        }


        //Структура що відповідає за зберігання інформації про питання,
        //яке висунуте на голосування
        struct VoteProposal {
          uint id; //Ідентифікатор пропозиції для голосування
          uint _type; // Тип операції, яка буде виконана після завершення голосування
          address creator;
          uint sharesVotesCount; // Кількість голосів "за"
          bool status; //Статус голосування (true - відкрите, false - закрите)

          //Додаткова інформація про голосування
          address addr;
          bytes32 name;
          bytes32 descr;
          uint share;
        }

        // Масив всіх питань, що висунуті на голосування
        VoteProposal[] voteProposals;
```

```
        // Функція для голосування "за"
        function vote(uint id) public {
          bool voteSuccess = false;
          bytes32 errMessage = "Vote ID is incorrect";
          for (uint i = 0; i < voteProposals.length; i++) {
            //Пошук питання що подано на розгляд
            if (voteProposals[i].id == id && voteProposals[i].status) {
              errMessage = "Cannot find voter";
              for (uint j = 0; j < shareholders.length; j++) {
                //Пошук інформації про людину, що голосує
                if (shareholders[j].account == msg.sender) {
                  //Додання її акцій до тих, що виступають "за"
                  voteProposals[i].sharesVotesCount += shareholders[j].share;
                  voteSuccess = true;
                  errMessage = "";
                }
                break;
              }
              break;
            }
          }
          //Емітування сигналу про те, що користувач успішно/неуспішно проголосував
          emit GotVote(id, msg.sender, voteSuccess, errMessage);
        }


        // Функція, що закінчує голосування та підбиває підсумки.
        // Може бути виконана лише особою, що подала питання на розгляд
        function finishVote(uint id) public {

          for (uint i = 0; i < voteProposals.length; i++) {
            //Пошук питання що подано на розгляд
                     if (voteProposals[i].id == id && voteProposals[i].status &&
voteProposals[i].creator == msg.sender) {
              //Закриття голосування
              voteProposals[i].status = false;
              //Перевірка результату голосування
              if (voteProposals[i].sharesVotesCount > getTotalShares() / 2) {
                //Емітування сигналу про те, що питання узгоджене
                emit VotingResult(id, true);
                //Введення рішення в дію
                if (voteProposals[i]._type == 0) {
                         addNewShareholder(voteProposals[i].name, voteProposals[i].descr,
voteProposals[i].addr, voteProposals[i].share);
                } else if (voteProposals[i]._type == 1) {
                  withdraw(voteProposals[i].share);
                } else if (voteProposals[i]._type == 2) {
                  destroyContract();
                } else if (voteProposals[i]._type == 3) {
                  transfer(voteProposals[i].addr, voteProposals[i].share);
                } else {
                  //Емітування сигналу про те, що питання не узгоджене
                  emit VotingResult(id, false);
                }
              } else {
                //Емітування сигналу про те, що питання не узгоджене
                emit VotingResult(id, false);
              }
              break;
            }
          }
        }

         //Функція для подання питання про додавання нового акціонера і виділення йому
акцій
          function addNewShareholderProposal(bytes32 name, bytes32 description, address
account, uint share) public {
```

```
          uint id = now;
          voteProposals.push(
            VoteProposal(id, 0, msg.sender, 0, true, account, name, description, share)
          );
          //Емітування сигналу про початок голосування
          emit NewVoting(id, msg.data);

        }

        //Функція для додавання нового акціонера і виділення йому акцій
         function addNewShareholder(bytes32 name, bytes32 description, address account,
uint share) private {
          shareholders.push(Shareholder(name, description, account, share));
          //Емітування сигналу про те, що додано нового акціонера
          emit ShareholderCreated(name, share, account);
        }

        //Функція для подання питання про розподіл дивідендів
        function withdrawProposal(uint amount) public {
          uint id = now;
          voteProposals.push(
            VoteProposal(id, 1, msg.sender, 0, true, msg.sender, "", "", amount)
          );
          //Емітування сигналу про початок голосування
          emit NewVoting(id, msg.data);

        }

        //Функція для розподілу дивідендів
        function withdraw(uint amount) private {
          //Перевірка можливості розподілу (Чи вистачає коштів)
          if (amount < address(this).balance) {
            uint totalShares = getTotalShares();
            for (uint i = 0; i < shareholders.length; i++) {
              //Розподіл коштів в залежності від частини акцій, якими володіє акціонер
                      shareholders[i].account.transfer(amount * shareholders[i].share /
totalShares);
            }
            //Емітування сигналу про успішний розподіл дивідендів
            emit WithdrawFinished(true, amount);
          } else {
            //Емітування сигналу про неуспішний розподіл дивідендів
            emit WithdrawFinished(false, amount);
          }
        }

        //Функція для подання питання про ліквідацію
        function destroyContractProposal() public {
          uint id = now;
          voteProposals.push(
            VoteProposal(id, 2, msg.sender, 0, true, msg.sender, "", "", 0)
          );
          //Емітування сигналу про початок голосування
          emit NewVoting(id, msg.data);
        }
        //Функція для подання питання про ліквідації контракту
        function destroyContract() private {
          //Розподіл всіх нерозподілених коштів
          withdraw(address(this).balance);
          //Емітування сигналу про ліквідацію
          emit ContractDestroyed();
              //Виклик дестуктора, перерахування залишкових коштів на адресу першого
акціонера
           //Так як використовується цілочисельне ділення то можуть залишитись незначні
залишки
          selfdestruct(shareholders[0].account);
        }
```

```
        //Функція для подання питання про переказ коштів на сторонній гаманець
        function transferProposal(address addr, uint amount) public {
          uint id = now;
          voteProposals.push(
            VoteProposal(id, 3, msg.sender, 0, true, addr, "", "", amount)
          );
          //Емітування сигналу про початок голосування
          emit NewVoting(id, msg.data);
        }
       //Функція для переказу коштів на сторонній гаманець
        function transfer(address addr, uint amount) private {
          //Перевірка можливості здійснення переказу
          if (amount < address(this).balance) {
            //Переказ
            addr.transfer(amount);
            //Емітування сигналу про успішний переказ коштів третій особі
            emit TransferResult(addr, amount, true, "");
          } else {
            //Емітування сигналу про неуспішний переказ коштів третій особі
            emit TransferResult(addr, amount, false, "Insufficent funds");
          }
        }

        //Функція для переказу акцій між акціонерами
        function transferShares(address to, uint amount) public {
          bool status = false;
          uint donorIdx;
          uint recepIdx;
          //Знаходження відправника
          for (uint i = 0; i < shareholders.length; i++) {
                 if (shareholders[i].account == msg.sender && shareholders[i].share >=
amount) {
              donorIdx = i;
              status = true;
            }
          }
          if (!status) {
               //Емітування сигналу про те, що в відправника немає достатньої кількості
акцій
           emit SharesTransfered(msg.sender, to, amount, false, "Insufficent number of
shares");
          } else {
            status = false;
            //Пошук отримувача
            for (i = 0; i < shareholders.length; i++) {
              if (shareholders[i].account == to) {
                recepIdx = i;
                status = true;
              }
            }
            if (!status) {
              //Емітування сигналу, що отримувача не знайдено
                     emit SharesTransfered(msg.sender, to, amount, false, "Cannot find
recepient");
            } else {
              //Переказ коштів
              shareholders[donorIdx].share -= amount;
              shareholders[recepIdx].share += amount;
              //Емітування сигналу про успішний переказ
              emit SharesTransfered(msg.sender, to, amount, true, "");
            }
          }
        }

      }
```