Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем

Реалізація за допомогою технології блокчейн системи винагороди експертів платформи Rateit.expert

Текстова частина до кваліфікаційної роботи

за спеціальністю «Комп'ютерні науки» - 122

Керівник кваліфікаційної роботи

старший викладач

Гороховський К.С.

_____ (підпис)

"___" _____ 2024 року

Виконав студент

КН-4 Яременко П. В.

"___" _____ 2024 року

Київ 2024

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем

ЗАТВЕРДЖУЮ

_____ старший викладач Гороховський К.С.

„_____” _____ 2023 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу

студенту Яременку Петру Всеволодовичу

факультету інформатики 4 курсу бакалаврської програми

ТЕМА: Реалізація за допомогою технології блокчейн системи винагороди експертів платформи Rateit.expert

Зміст ТЧ до кваліфікаційної роботи:

Індивідуальне завдання

Вступ

Розділ 1. Дослідження та аналіз предметної області

Розділ 2. Аналіз технічного завдання

Розділ 3. Розробка застосунку

Висновки

Список використаної літератури

Додатки (за необхідністю)

Дата видачі „___” _____ **2023 р.** **Керівник**_____

(підпис)

Завдання отримав_____

(підпис)

Календарний план виконання роботи

№	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на кваліфікаційну роботу	01.10.2023	
2.	Огляд літератури за темою роботи	01.11.2023	
3.	Проведення дослідження	01.12.2023	
4.	Написання програмного застосунку	01.01.2024	
5.	Написання текстової частини	04.04.2024	
6.	Захист кваліфікаційної роботи		

Студент _____

Керівник _____ “_____”_____ 2023р.

Зміст

Перелік умовних позначень

DAO - децентралізована автономна організація.

HIPAA - акт про мобільність та підзвітність медичного страхування.

GDPR - загальний регламент регуляції безпеки даних.

SOL - нативний токен блокчейн мережі Solana.

SDLC - життєвий цикл розробки програмного забезпечення.

Анотація

Кваліфікаційна робота присвячена проблемам колаборативної розробки документів та розподілу винагород між розробниками. За метод вирішення проблеми було обрано реалізацію інструменту колаборативної розробки та імплементація DAO в систему з функціоналом мультипідписного гаманця.

Розроблено програмний застосунок для колаборативної розробки документів в форматі ".md". Досліджено та імплементовано DAO та мультипідписні гаманці, як підхід до вирішення задачі розподілу винагород, також написано власний алгоритм.

Текстова частина курсової містить опис дослідження проблем колаборативної розробки та доцільності використання вищезазначених технологій для досягнення поставлених цілей. Текстова частина курсової також описує необхідні теоретичні відомості про використані інструменти та технології.

Ключові слова: *блокчейн, DAO, multisig, Solana, Squads, колаборативна розробка, NestJS.*

Вступ

Актуальність теми

Актуальність дослідження обумовлена зростаючою потребою в колаборативній розробці документів у сучасному світі, де дедалі більше проектів реалізується командно. Ефективні інструменти для спільної роботи значно підвищують продуктивність і якість результатів, забезпечуючи можливість синхронної роботи команд.

Важливим аспектом є також розподіл винагород між розробниками у багатокористувацьких проектах. Забезпечення справедливого розподілу винагород між членами команди за допомогою децентралізованих автономних організацій (DAO) та мультипідписних гаманців сприяє прозорості і справедливості цього процесу, стимулюючи активну та якісну роботу учасників. Інноваційність підходу полягає в інтеграції DAO та мультипідписних гаманців у систему. Це новаторське рішення поєднує передові технології з реальними потребами управління командною роботою, відкриваючи нові можливості для автоматизації та оптимізації процесів.

Практична значущість дослідження полягає у розробці програмного застосунку для колаборативної розробки документів у форматі ".md" та імплементації сучасних технологій для розподілу винагород. Такий застосунок може бути використаний у різних галузях, від інформаційних технологій до наукових досліджень, де колаборативна робота є ключовим елементом успіху.

Таким чином, дослідження є актуальним і значущим не лише з наукової точки зору, але й з практичної, сприяючи розвитку інструментів колаборативної роботи та справедливого розподілу винагород.

Об’єкт дослідження

Колаборативна розробка документів та розподіл винагород за даний процес.

Предмет дослідження

Імплементація DAO та мультипідписних гаманців для реалізації розподілу винагород та гарантування прозорості даних.

Мета дослідження

Визначити, чи можливе використання DAO та мультипідписного гаманця для забезпечення прозорості даних та реалізації розподілу винагород під час колаборативної розробки технічних документів.

Постановка задачі

1. Аналіз засобів для колаборативної розробки документів.
2. Дослідження DAO.
3. Дослідження мультипідписного гаманця.

4. Розробка власної платформи для колаборативної розробки документів із закладенням можливостей для розширення системи.

Структура роботи:

1. В першому розділі викладено загальні відомості про предметну область, описано наявні проблеми та запропоновано, та обґрунтовано підхід до їх вирішення. Крім цього, проаналізовано існуючі рішення на ринку.
2. В другому розділі проаналізовано технічне завдання: описано користувачів, функціональні вимоги та вимоги до даних.
3. В третьому розділі обґрунтовано вибір інструментів, детально описано реалізацію ключових компонентів системи та наведено результати роботи програми.

РОЗДІЛ 1. Дослідження та аналіз предметної області

1.1. Загальні відомості

Колаборативні засоби розробки є невід’ємним інструментом при створенні програмного забезпечення. З розвитком індустрії програмування фактично в усьому циклі розробки присутня команда щонайменше з 4-5 позицій різної царини відповідальності [1]. Таким чином середньостатистичний склад розробників на web3 проекті містить такі позиції: backend-blockchain розробник, frontend розробник, дизайнера архітектора та бізнес аналітик (може варіюватись залежно від специфіки проекту) [2].

Задля підвищення ефективності та оптимізації часу та ресурсів прийнято використовувати колаборативні засоби розробки, які значно спрощують взаємодію, співпрацю і управління проектами між членами команди, незалежно від їхнього місцезнаходження. Ці інструменти забезпечують спільну роботу над кодом, документацією, задачами та іншими аспектами розробки.

За даними джерела “Statista” ринок колаборативних інструментів США становить близько 15,05 мільярдів доларів США станом на 2024 рік та очікує сукупний середньорічний зріст (GAGR) на 1,73% до 2028 року. В Сполучених Штатах ринок програмного забезпечення для співпраці контролюється кількома основними гравцями, такими як Microsoft Teams та Slack, які складають велику частину ринку. Однак спостерігається також зростаюча тенденція до спеціалізованих інструментів, які відповідають конкретним галузям або сценаріям використання. Наприклад, існують інструменти для співпраці, розроблені спеціально для галузі охорони здоров'я, які відповідають вимогам HIPAA. У Європі на ринку програмного

забезпечення для співпраці спостерігається зростання попиту на інструменти, що спрямовані на забезпечення конфіденційності та відповідають вимогам GDPR. Це призвело до появи нових колаборативних засобів розробки, які надають пріоритет конфіденційності та безпеці даних [3].

Очікується, що ринок програмного забезпечення для співпраці в усьому світі буде продовжувати зростати в найближчі роки через збільшення прийняття віддаленої роботи та зростання економіки. Пандемія COVID-19 також прискорила впровадження інструментів співпраці, оскільки бізнеси мусили адаптуватися до віддаленої роботи у великому масштабі. Загалом ринок програмного забезпечення для співпраці є високо конкурентним та постійно еволюціонує, щоб задовольнити змінні потреби бізнесу та команд, адже існує величезна потреба у рішеннях для віддаленої роботи та зростаюча популярністю хмарних сервісів. Клієнти вимагають більш інтегрованих та зручних у використанні інструментів для співпраці, до яких можна отримати доступ з будь-якого місця і в будь-який час та які допоможуть зберегти зв'язок та продуктивність команд.

1.2. Проблематика

Попри всебічну популярність колективного підходу до розробки, величезна кількість фахівців стикається з проблемами координації розподіленої розробки проекту, комунікації між командами та узгодженості задачі [4]. Також, основною проблемою є кваліфікація фахівців, які виконують ті, чи інші функції в команді. Як ми бачимо з досвіду великих компаній, SDLC постійно потребує рефакторингу та циклічних підходів до розробки. Саме для цього були створені такі методології, як SCRUM та

AGILE, в яких закладено циклічну компоненту, яка й забезпечує якість виконання тих чи інших задач [5].

Також досить вагомою проблемою при колаборативній розробці стає різний рівень компетентності фахівців та узгодженості їхніх рішень. З огляду на це в тому числі якраз і виникає потреба в циклічних підходах. Уявімо, що першопочатково при створенні того чи іншого продукту була можливість розробити рішення, узгоджене всією командою, де знаходяться лише відібрані фахівці. Очевидно, що ефективність зросла б, а витрати зменшились з рахунок того, що більшість проблемних питань будуть виявлені та пропрацьовані з самого початку.

1.3. Аналіз наявних рішень

На даний момент існує декілька основних гравців на ринку застосунків для колективної розробки, а саме Google Docs, Microsoft 365, GitHub, Confluence, Jira і тд. (треба ще статки накидать).

На жаль, жоден з них не надає зручного функціоналу для узгодження рішень всією командою хоча б через те, що вхідний документ не декомпозується на логічні складові та не вимагає консенсусної оцінки кожної компоненти. Першопочатково фактично всі існуючі засоби пропонують модель, при якій існує певний фахівець, який розробляє стартовий документ, або чернетку під нього і дає його на перевірку компетентній особі. Саме тому й доводиться використовувати методології та перероблювати одне й те саме, витрачаючи при цьому велику кількість часових та фінансових ресурсів.

1.4. Опис обраного підходу до вирішення поставленої задачі

Якщо заглибитись в проблематику та проаналізувати наявні на ринку рішення, то очевидним стає факт того, що існує необхідність в імплементації децентралізованого підходу всередину класичних методів розробки тих, чи інших продуктів. На нашу думку, що створення розподіленої системи, яка не буде потребувати модератора якраз і є однією із задач, які можуть бути ідеально вирішені за допомогою технології блокчейн [6].

Враховуючи, що ідеологічно дана технологія першопочатково пропонує прозорість та децентралізованість саме вона стала основою для вирішення даної проблеми. Для перевірки даного підходу та його імплементації було вирішено обрати процес створення технічних документів README, на базі якого було імплементовано логіку децентралізації. Було запропоновано автоматично розбивати вхідний документ на відділені одна від одної компоненти, які фактично стають основними вузлами та об’єктами редагування за допомогою генеративного претренованого трансоформеру ChatGPT-4. В свою чергу фахівці, що беруть участь в розробці документу повинні оцінити кожну з компонент для їх консенсусного затвердження. Після затвердження усіх компонент документ зберігається.

Також однією з ключових задач було забезпечення унеможливлення фальсифікації оцінок, підміни самих компонент та реалізації розподілу винагород серед експертів, які беруть участь в розробці.

1.4.1. Аналіз СКОРАД та його недоліків

Свого часу студентка НаУКМА Анастасія Новохацька проводила дослідження в схожій царині, та розробляла застосунок зі схожою логікою,

який був взятий за основний прототип новоствореної системи RateIT [7]. У вищезгаданому сервісі була виявлена низка недоліків, які певною мірою унеможливлювали його публічне використання та можливість монетизації роботи експертів, а саме:

1. Неефективна архітектура застосунку та його реалізація, яка унеможливлювала імплементацію розподілу винагород за роботу та забезпечення відсутності фальсифікацій оцінок фахівців та самих компонент.
2. Перевантажена користувацька логіка.
3. Відсутність автоматичної декомпозиції вхідного документу на самостійні за логікою компоненти.

Саме тому було прийняте рішення модифікувати дану систему під сучасний ринок, використовуючи сучасні методи розробки та архітектурні рішення.

1.4.2. Аналіз переконфігурації СКОРАД у RateIT та перехід від Web2 до Web3

Для досягнення цілей, для яких виконувалася дана робота було вирішено перейти з Web2 на Web3 архітектуру, адже серед зазначених друга унеможливлює імплементацію технології блокчейн. Web2, яка розвивалася на початку 21-го століття, залежить від централізованих платформ, які контролюють доступ користувачів та дані. Такі платформи, як Google та Facebook, керують та монетизують дані користувачів, часто з обмеженою прозорістю щодо того, як ці дані використовуються та як ними діляться. Саме ця централізація й приводить до проблем, таких як цензура,

порушення безпеки даних та обширне спостереження за користувачами з боку корпорацій та урядів [8].

На противагу цьому, Web3 представляє собою децентралізовану архітектуру, де контроль розподіляється серед користувачів, а не концентрується в руках кількох корпорацій. Такий підхід використовує блокчейн для забезпечення того, що власність над даними залишається у користувачів, покращуючи приватність та безпеку. Децентралізований підхід, який закладено в ідеологію Web3 робить його несхильним до цензури та централізованого контролю, сприяючи створенню більш відкритого та орієнтованого на користувача середовища. Web3 також дозволяє різним платформам спілкуватися більш ефективно та дає користувачу мати єдину онлайн-ідентичність на різних сервісах. Цей перехід до децентралізації обіцяє більше прав і можливостей для користувачів, де люди контролюють свої цифрові ідентичності та транзакції без посередників. Більше того, Web3 дозволяє проводити прямі однорангові транзакції за допомогою криптовалют, прибираючи залежність від традиційних фінансових систем і знижуючи вартість транзакцій. [9]

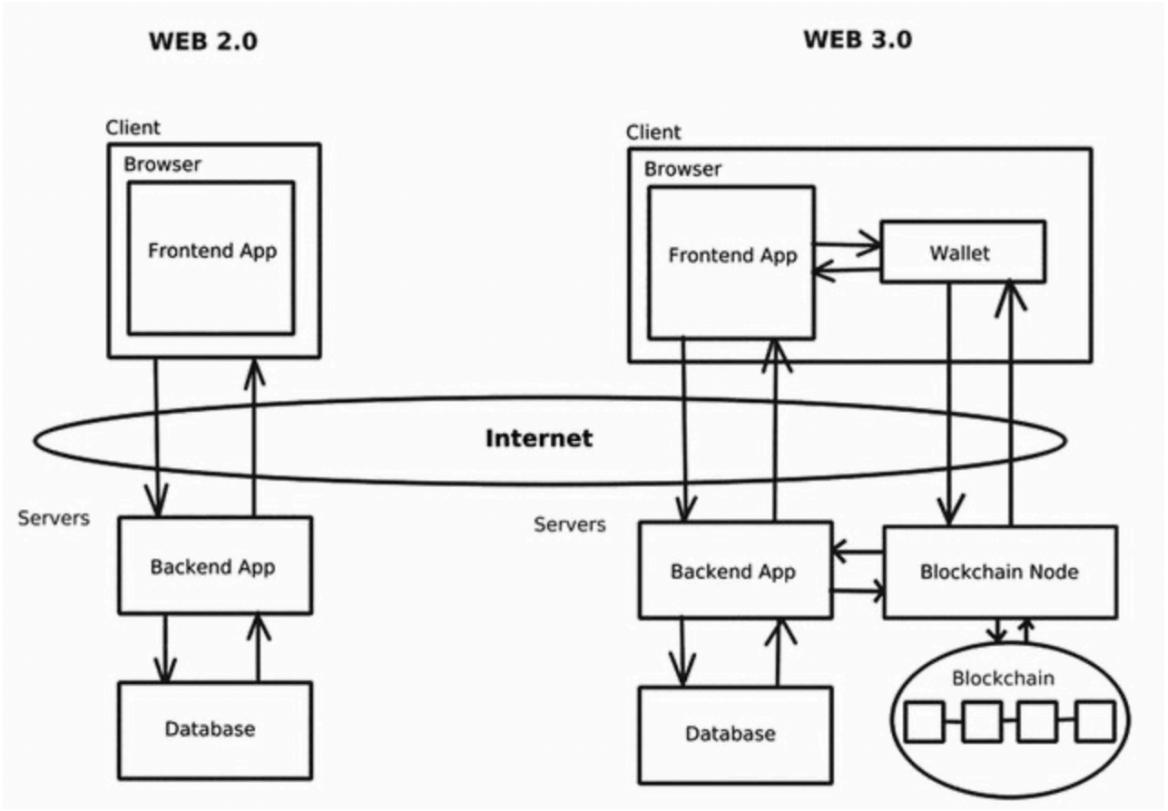

Рис. 1.4.2.1. *Порівняння архітектур Web2 та Web3 підходу* [9].

1.5. Обгрунтування вибору рішення

1.5.1. Обгрунтування вибору блокчейн протоколу Solana та його екосистеми

Як основу для реалізації блокчейн логіки було обрано Solana, яка є однією з передових платформ блокчейну і широко визнана завдяки своїй швидкості, ефективності та привабливості для розробників децентралізованих додатків (dApps).

Також проектуючи систему було вирішено обирати технології із запасом для масштабування, саме цим і зацікавила Solana, адже завдяки алгоритму консенсусу Proof of History (PoH), Solana дозволяє підтримувати високий обсяг транзакцій без втрати продуктивності. Вона може обробляти

понад 65,000 транзакцій на секунду (TPS), що значно перевищує більшість інших блокчейнів, таких як Ethereum, тому це забезпечує майже миттєву обробку даних, що є критично важливим для багатьох застосунків. Також вартість транзакцій у мережі Solana є дуже низькою порівняно з іншими блокчейн платформами, що робить її економічно вигідною для розробників і користувачів.

Solana використовує PoH разом із Proof of Stake (PoS), що допомагає вирішити проблему "порядку транзакцій", забезпечуючи швидшу валідацію та зменшення можливості маніпуляцій з транзакціями [10].

Окрім цього, одним із найважливіших факторів при виборі технологій стала екосистема. За останні декілька років Solana розрослась до достатньо великих масштабів, які забезпечують велику кількість постійних оновлень та нових продуктів, завдяки чому можна знайти ідеальне рішення під свою конкретну задачу.

1.5.2. Обгрунтування необхідності використання мультипідписних гаманців та створення відокремлених DAO.

Для реалізації вимог узгодженості та розподілу винагород було прийнято рішення імплементувати DAO та мультипідписні гаманці, адже вони відіграють важливу роль у забезпеченні безпеки та ефективного управління. DAO представляють нове покоління організацій, що вписуються в концепцію "Суспільства 5.0" - суспільства, де людина та технології тісно взаємодіють для вирішення соціальних проблем та покращення якості життя. Завдяки децентралізованій структурі та демократичному управлінню, DAO мають потенціал вплинути на різні аспекти суспільства, включаючи економіку, культуру та соціальну сферу.

DAO функціонує за допомогою смарт-контрактів, які дозволяють її учасникам приймати рішення колективно, голосуючи за пропозиції та управлінські питання та вирізняється такими ключовими характеристиками, як децентралізація та автономність, прозорість та безпека, токенізація, а також різноманітність цілей. DAO працює без центрального органу управління, дозволяючи кожному учаснику впливати на рішення шляхом голосування. Всі дії організації зберігаються в блокчейні, що забезпечує їх прозорість та унеможливлює зміну заднім числом. Учасники DAO часто володіють токенами, які надають їм права голосу та можуть використовуватися для управління організацією. Організації цього типу можуть мати різні цілі, від управління протоколами DeFi до колекціонування NFT та підтримки соціальних ініціатив. Тому, у контекстах, що вимагають консенсусу щодо чутливих або значущих рішень, таких як формування політик або розподіл ресурсів у спільнотних проєктах, DAO можуть демократизувати прийняття рішень і збільшити залученість зацікавлених сторін [11].

За рахунок того, що всі транзакції та рішення в рамках DAO записуються у блокчейн, це робить їх незмінними та легко аудитованими будь-якою стороною. Ця прозорість гарантує, що всі члени мають доступ до однакової інформації та можуть перевірити правильність та справедливість всіх дій. Вроджені особливості безпеки блокчейну захищають від різних ризиків, включаючи несанкціонований доступ та шахрайство. Оскільки DAO розподіляють дані між кількома вузлами, вони менш вразливі до кібератак, які зазвичай використовують одну точку відмови в централізованих системах. Більше того, механізми консенсусу, які використовуються в блокчейн платформах, вимагають валідації

декількома вузлами, що додатково забезпечує безпеку мережі від зловмисного втручання і записує лише дійсні транзакції.

Використання мультипідписних гаманців дозволяє досягти більшої прозорості та зменшити ризики, пов'язані з управлінням коштами в DAO. Однією з головних переваг мультипідписного гаманця є те, що для здійснення транзакцій потрібна згода певної кількості учасників. Це значно ускладнює несанкціонований доступ або крадіжку коштів, оскільки потенційний зловмисник мусить компрометувати декілька приватних ключів одночасно. Також для нас це цікаво й з іншого боку, адже мультипідписний гаманець дає змогу розподілити кошти, які були переведені на нього серед учасників даного мультипідпису, за рахунок цього ми можемо децентралізувати та зробити автономним один з основних процесів, а саме розподіл винагороди.

Також, мультипідпис зменшує ризики, пов'язані з помилковими транзакціями або зловживаннями з боку одного адміністратора. Використання мультипідпису децентралізації владу в DAO, розподіляючи відповідальність за управління коштами між декількома сторонами. Це відповідає основній ідеології DAO – забезпечити владу спільноті, а не окремим особам.

В нашому випадку, мультипідписні гаманці в DAO не просто додають шар безпеки, але й сприяють втіленню основних принципів децентралізації та колективного управління, які є ключовими для успішного функціонування децентралізованих автономних організацій, та безпосередньо нашого застосунку. Використання даного рішення дозволяє кластеризувати групи користувачів на окремі DAO, що безперечно розширює можливості даної системи.

1.5.3. Обгрунтування вибору технології Squads DAO

Для створення персоніфікованих DAO всередині системи під кожну команду було обрано Squads, який є безперечно одним з лідерів в цій галузі. Дана технологія була написана на базі блокчейну Solana, та активно з ним інтегрується. Основний функціонал, який ми використали - це власне створення та координація самого DAO, та мультипідписний гаманець.

Squads DAO дозволяє користувачам встановлювати різні рівні прав та обмежень для членів організації, включаючи різні моделі голосування та ухвалення рішень. Це допомагає адаптувати DAO до специфічних потреб і структур керування кожної команди або проекту. Як і більшість DAO, Squads сприяє високому рівню прозорості у фінансових операціях та ухваленні рішень. Всі транзакції та голосування документуються на блокчейні, що забезпечує чесність та доступність інформації для всіх членів [12].

Також, Squads DAO може інтегруватися з іншими блокчейн-сервісами та платформами, забезпечуючи ширші можливості для розширення та розвитку DAO.

1.5.4. Обгрунтування вибору DiscordAPI.

Задля зручності користування постало необхідним питання реалізації категоризації документів та їхня кластеризація по різним командам, DAO, для цього було вирішено використати Discord, як основну платформу, команди та категорії з якої можна перевикористовувати для нашої потреби, адже детально проаналізувати активність крипто-спільноти серед різних засобів комунікації, то стає очевидним раціональність використання даної платформи [13].

Фактично, за рахунок цього ми можемо покласти шар логіки, пов’язаний зі створенням команд та категорій на вже готовий продукт, який активно використовується спільнотою. Таким чином, відбувається створення додаткової реляції, між двома різними платформами, яке дозволяє взаємодіяти з однією всередині іншої. Задля цього було запропоновано реалізувати бота, який би мав доступ до серверу в сервісі Discord, та міг би комунікувати з його читачами щодо оновлень в нашій системі.

РОЗДІЛ 2. Аналіз технічного завдання

Головна мета застосунку:

Надати можливість декомпозованої колективної розробки технічних документів в форматі Markdown користувачам з консенсусом щодо фінальних компонентів документу. Гарантувати прозорість та відсутність фальсифікованих оцінок чи компонент документу за допомогою блокчейну та персоналізованих DAO.

2.1. Опис ролей в системі

На даному етапі існує дві основних ролі - Product Owner та Editor, які одночасно присвоєму кожному користувачу.

2.2. Технічні вимоги до ролей, набутих користувачем системи.

Product owner:

- створити документ в форматі .md та відповідне для нього DAO
- створити документ в форматі .md та відповідне для нього DAO
- внести депозит, як гарантію винагороди за розробку документу
- переглянути категорії та сервери, до яких є доступ на платформі
- переглянути свої документи
- автоматично розподілити винагороду між редакторами
- фіналізувати документ

Editor:

- створювати редакції, до компонент документу
- створювати нові компоненти документу
- коментувати компоненти документу

2.3. Специфікації вимог до даних

Про користувачів системи мають зберігатися наступні дані: унікальний ідентифікатор (Discord ID), логін (Discord username), адреса криптовалютного гаманця.

Про документ мають зберігатися наступні дані: унікальний номер, назва, DiscordID власника, адреса мультипідписного гаманця розробників, які будуть брати участь в розробці.

Про компоненту документу мають зберігатися наступні дані: унікальний номер документу, рейтинг, булеве значення чи є компонента затвердженою, текст, який в ній міститься.

РОЗДІЛ 3. Розробка застосунку

3. Опис загальної архітектури.

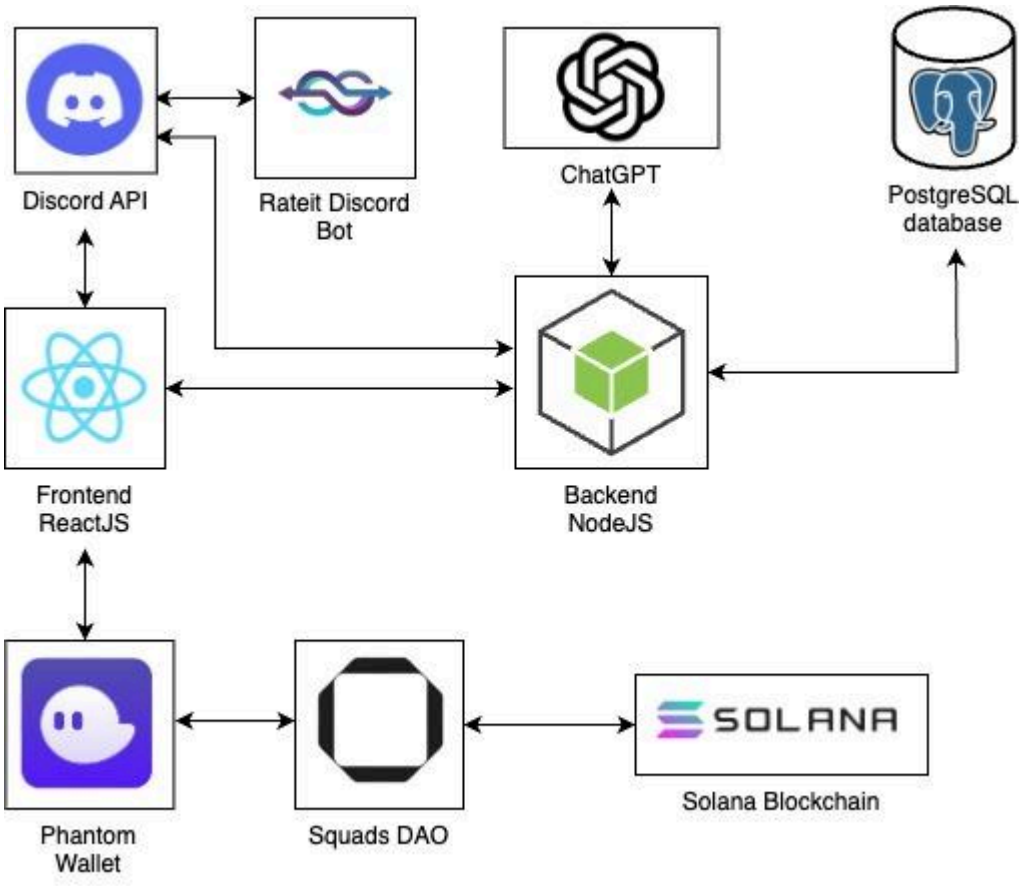

Рис. 3.1. *Загальна архітектура застосунку.*

Якщо розглянути загальну архітектуру застосунку, то стає видно, що було обрано класичний підхід розробки децентралізованих застосунків та інтегровано багато сервісів третіх сторін. Такий підхід зумовлений ефективністю використання вже готових рішень, які можуть покрити велику кількість логіки, яку необхідно реалізувати задля функціонування системи.

В цілому в нас є клієнт, який перенаправляє запити на сервер та на смарт-контракт в блокчейн. При обробці документу ми створюємо

транзакцію в блокчейн, за допомогою якої створюємо DAO під документ, далі сервер обробляє документи за допомогою ChatGPT-4 та записує їх в базу даних, потім за допомогою DiscordAPI надсилає повідомлення в бот, який в свою чергу перенаправляє його вже на сервер, де розміщений документ, після цього на клієнт повертається відображається вже оброблений документ у вигляді JSON-об’єкту, який відображається за допомогою вбудованих у бібліотеку ReactJS методів.

3.1. Основні засоби розробки та їхнє використання.

Для розробки даної платформи було обрано одні з найпопулярніших фреймворків та засобів розробки, які є на сучасному ринку.

3.1.1. NestJS.

NestJS є фреймворком для Node.js, який надає можливість розробникам легко створювати ефективні, надійні та легко масштабовані серверні застосунки. NestJS використовує модульну структуру, яка дозволяє організовувати код за допомогою чітко відокремлених модулів. Це спрощує управління залежностями і розширення програм.Також NestJS використовує систему впровадження залежностей (Dependency Injection), що дозволяє легко управляти залежностями між класами і інстанціями. Основна ідея даного підходу полягає в тому, що компоненти або об'єкти не повинні самостійно створювати залежності (інші об'єкти, які вони використовують). Натомість, ці залежності мають бути "введені" в об'єкт ззовні, зазвичай через конструктор, сетери, або інтерфейси.

Також NestJS має вбудовану підтримку для створення мікросервісних архітектур, включаючи взаємодію через різні транспортні шари, такі як TCP, MQTT чи WebSocket, що дає нам велику переваги, якщо буде потреба в інтеграції розробленого застосунку всередину іншого.

Також досить важливим аспектом при виборі стали численні пакети та інтеграції для розширення функціональності NestJS, включаючи вбудовану TypeORM, можливість реалізації аутентифікації за допомогою Passport та багато іншого [14].

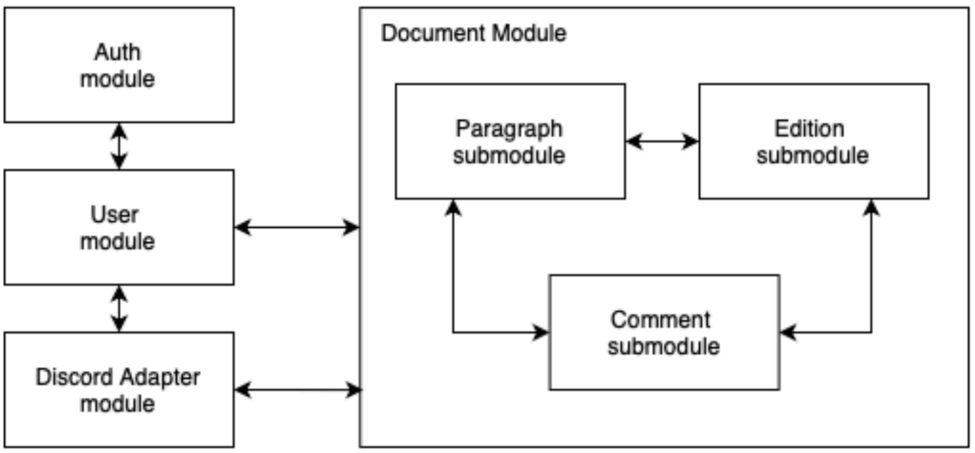

Рис. 3.1.1.1. *Архітектура модулів серверної частини.*

Задля реалізації бізнес логіки було розроблено модульну архітектуру серверної частини, яка буде забезпечувати можливість подальшого масштабування системи. Фактично, ми створюємо модуль під кожен шар бізнес-логіки та пишемо для нього імплементацію, інкапсулюючи модулі одне від одного. Переваги даного підходу абсолютно очевидні, адже правильно спроектовані модулі мають між собою низьку консистентність, тим самим не перенавантажуючи потоки даних під час своєї взаємодії. Попри це є певний недолік, який стосується продуктивності даної архітектури, але в нашому випадку це не настільки важливо, як можливість легкої подальшої інтеграції в інші застосунки [15].

Обраний фреймворк дає змогу виокремити всю бізнес-логіку у окремий мікросервіс, який можна буде інтегрувати в інші продукти.

3.1.2. Discord API.

Discord API дозволяє розробникам інтегрувати ботів або сервіси з платформи Discord в інші застосунки, використовуючи різні можливості для маніпуляції серверами, каналами, що дозволило нам реалізувати команди та категорії документів в середині команд, використовуючи нативні сутності (Guilds, Channels). В нашому випадку найцікавішим було створення боту, який буде надавати можливість інтеграції створеного сервісу у вже існуючі спільноти, що спрощує використання платформи RateIT, робить її більш дружньою для користувачів та дає можливість комунікації між двома додатками [16].

Також за допомогою даного сервісу була реалізована логіка автентифікації, а саме OAuth2, за допомогою імплементованих під Discord вбудованих в NestJS passport-стратегій. За рахунок даного функціоналу ми даємо змогу користувачам безпечно надавати доступ до свого Discord акаунту без необхідності передавати свої облікові дані.

3.1.3. Solana Program Library.

Бібліотека програм Solana (SPL) — це набір вбудованих програм, розроблених для надання широкого спектру функціональних можливостей розробникам, які створюють застосунки на блокчейні Solana. SPL є ключовим елементом у розширенні можливостей блокчейну Solana та наданні розробникам попередньо побудованих функціональних можливостей, які прискорюють процес розробки децентралізованих застосунків.

Використання даної бібліотеки було необхідним для того, щоб взаємодіяти з блокчейном та контрактами Squads, які були використані для реалізації DAO, Multisig та розподілу винагород [17].

3.1.4. SquadsV4.

Squads Protocol - це платформа, реалізована для створення та керування мультипідписними гаманцями, реалізована на блокчейні Solana. Цей протокол дозволяє створювати та керувати гаманцями, які підтримують складні функціональні можливості, такі як мультипідпис, абстракція облікових записів та програмовані ліміти витрат.

Основні особливості Squads Protocol включають його використання логіки мультипідпису, яка вимагає підтвердження транзакцій кількома сторонами (governance & voting), що підвищує безпеку цифрових активів. Крім того, платформа підтримує абстракцію облікових записів, що дозволяє більш гнучке та програмоване управління операціями гаманця порівняно з традиційними гаманцями, відділяє власника облікового запису від підписувача транзакцій [12].

3.1.5. PostgreSQL, TypeORM.

При виборі системи управління базою даних вибір впав на PostgreSQL, адже ця система відома своєю надійністю, гнучкістю та підтримкою великої кількості типів даних, що робить її популярною для складних застосунків. PostgreSQL підтримує різноманітні типи даних, включаючи геопросторові дані та JSON, забезпечує транзакційну цілісність за допомогою ACID-сумісності та підтримує складні SQL-операції, такі як підзапити, тригери та в'юхи. Система може бути налаштована за допомогою розширень, що дозволяє створювати нові функції та типи даних [18].

В якості об’єктно-реляційного маппера (ORM) було використано TypeORM. Ця бібліотека підтримує такі функції, як ліниве та жадібне завантаження асоціацій, підтримку транзакцій, міграції баз даних,

автоматичне створення схем та складні запити через репозиторії або активні записи [19].

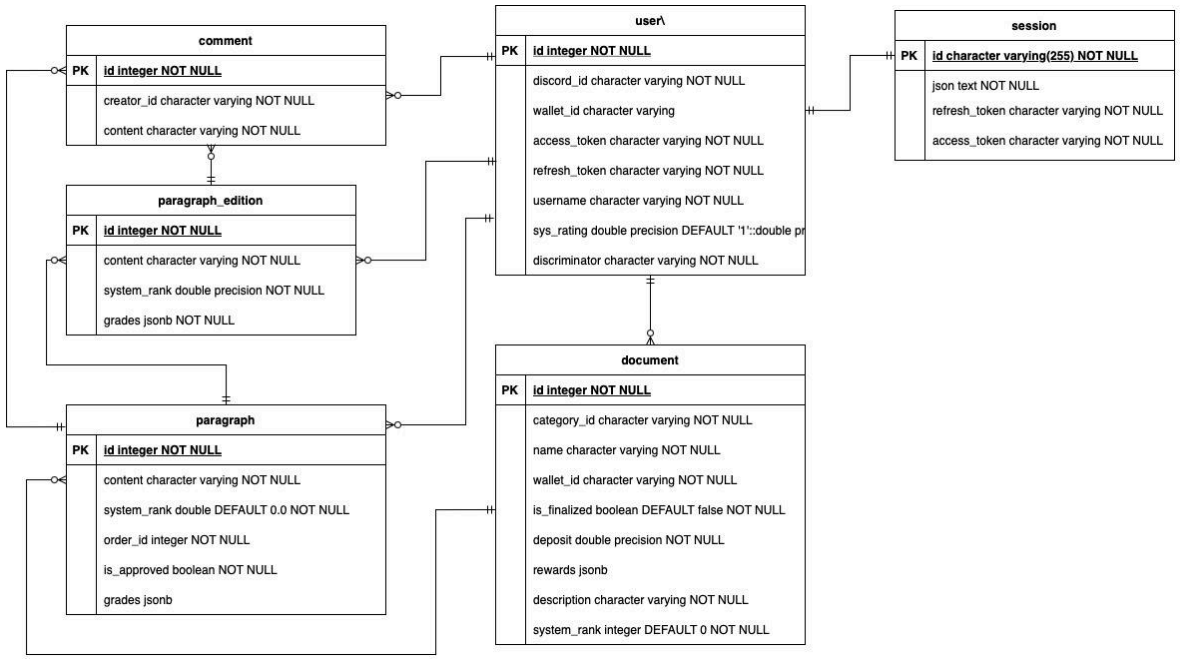

Рис. 3.1.5.1. *ER-модель бази даних.*

3.1.6. Інші засоби розробки.

ReactJS:

Для розробки користувацького інтерфейсу був обраний ReactJS, завдяки своїм перевагам, які включають компонентну архітектуру, декларативний підхід, використання віртуального DOM. Компонентна структура React дозволяє створювати незалежні та перевикористовувані частини інтерфейсу, що спрощує та пришвидшує подальшу розробку та підтримку, адже зміни в одній компоненті не впливають на решту системи, що є важливою перевагою в управлінні проектами.

React має “під капотом” використовує віртуальний DOM, який дозволяє оптимізувати оновлення реального DOM, значно підвищуючи

продуктивність застосунків, особливо при роботі з великими обсягами даних.

Враховуючи ці переваги, вибір ReactJS для розробки UI частини є стратегічним рішенням, яке підвищує якість та швидкість розробки, забезпечуючи при цьому масштабованість і гнучкість проекту.

Docker:

Для пакування реалізованого програмного застусунку разом з усіма залежностями в стандартні одиниці, було використано Docker, який працює за допомогою технології контейнеризації, що дозволяє ізолювати додатки в контейнерах, забезпечуючи консистентність середовища незалежно від місця їх виконання. Docker використовує функції ядра Linux, такі як cgroups та namespaces, для ізоляції ресурсів і створення безпечного виконавчого середовища для кожного контейнера. Він має центральне ядро (Docker Daemon), яке керує усіма аспектами контейнеризації, від створення та управління образами контейнерів до керування їх виконанням. Docker Engine працює на хостовій машині та взаємодіє з ядром операційної системи для створення та управління контейнерами.

Також було використано інструмент Docker Compose, який дозволяє визначати та запускати багатоконтейнерні додатки всередині Docker, який використовує YAML файл для конфігурації сервісів, обсягів та інших залежностей, що спрощує управління складними додатками. Ця архітектура забезпечує гнучкість та масштабованість, дозволяючи легко розгортати, оновлювати та масштабувати додатки за допомогою контейнерів. Завдяки контейнерам, можна мінімізувати проблеми, пов'язані з відмінностями між розробницькими та продукційними середовищами, та

зосередитися на вдосконаленні функціональності та продуктивності додатку.

3.2. Використання DAO та Multisignature wallet, які пропонує сервіс Squads.

Для створення мультипідписного гаманця було обрано ті, які пропонує Squads Protocol. Дана реалізація криптовалютного гаманця використовує TSS (Threshold Signature Scheme). Саме це дозволяє групі розробників документу згенерувати цифровий підпис таким чином, що тільки певна підмножина цих розробників (кількість яких визначається пороговим значенням (threshold) може підписати повідомлення. Наприклад, у схемі *(t, n)* - порогового підпису секретний ключ розділяється на *n* частин, і для створення підпису необхідно щонайменше *t* учасників. Якщо кількість учасників, що беруть участь у підписанні, менша за *t*, вони не можуть генерувати дійсний підпис. В нашому випадку для максимальної узгодженості $t = n$, тобто для будь-якої дії запису в блокчейн підписати мають всі учасники.

На першому етапі головний секретний ключ генерується центральним органом або довіреною третьою стороною і розподіляється між усіма учасниками системи. Цей процес зазвичай здійснюється за допомогою спеціальних алгоритмів, таких як шифрування Шаміра. Для підпису повідомлення кожен з t учасників використовує свою частину секретного ключа для створення часткового підпису. Потім ці часткові підписи об'єднуються, щоб створити повний підпис, який можна перевірити за допомогою загальнодоступного ключа. Повний підпис

перевіряється звичайним способом за допомогою загальнодоступного ключа, що гарантує цілісність та автентичність підписаного повідомлення.

TSS забезпечує високу стійкість до компрометації, оскільки для отримання повного секретного ключа необхідно володіти певним мінімальним числом часткових ключів. Навіть якщо деякі часткові ключі будуть скомпрометовані, підпис залишиться безпечним. У разі виходу з ладу деяких учасників система залишається функціональною, якщо залишилося щонайменше *t* учасників, здатних підписати повідомлення. Це робить TSS стійкою до відмов. [20]

3.3. Реалізація розподілу винагород.

Для розподілу винагород був розроблений алгоритм, який бере кількість змін, які були внесені розробником, та обраховані протягом роботи над документом, тобто кількість його коментарів, редакцій та оцінок, які він залишив в документі. Дане значення переводиться у відсоток від всього депозиту, внесеного в документ та множиться на власне суму депозиту. Таким чином винагорода розподіляється лише на тих розробників, котрі працювали над розробкою.

```
user_grade_count_map = [{username: value}]
reward_distribution = []

for each array user_grade_cortage in array all_grades_array:
  for each object grade in user_grade_cortage:
        get username from grade
        if username is in user_grade_count_map:
            increment the value of user_grade_count_map for username by 1
        else:
            set the value of user_grade_count_map for username to 1

for each username in user_grade_count_map:
    get count from user_grade_count_map for username
    calculate result as deposit divided by count
    set result for username in reward_distribution

return reward_distribution
```

Рис. 3.3.1. *Псевдокод запропоноваго алгоритму.*

3.4. Тестування застосунку та результати роботи.

Першочерговою задачею для початку роботи є авторизація.

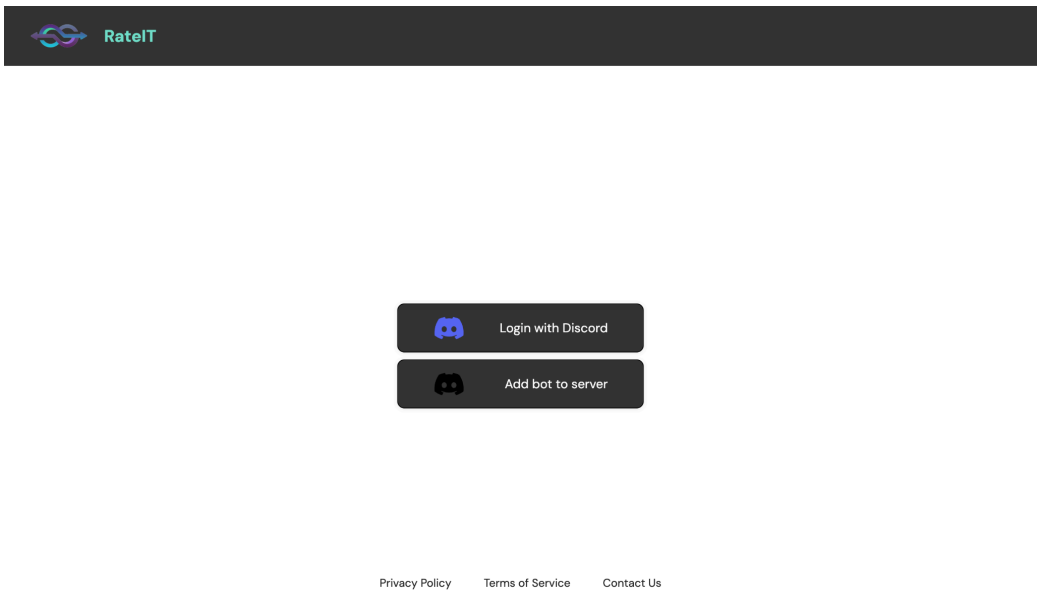

Рис. 3.4.1. *Сторінка авторизації.*

Після авторизації будуть показані сервери, та їхні категорії, в яких є бот. Також є опція додати бот до нового сервера.

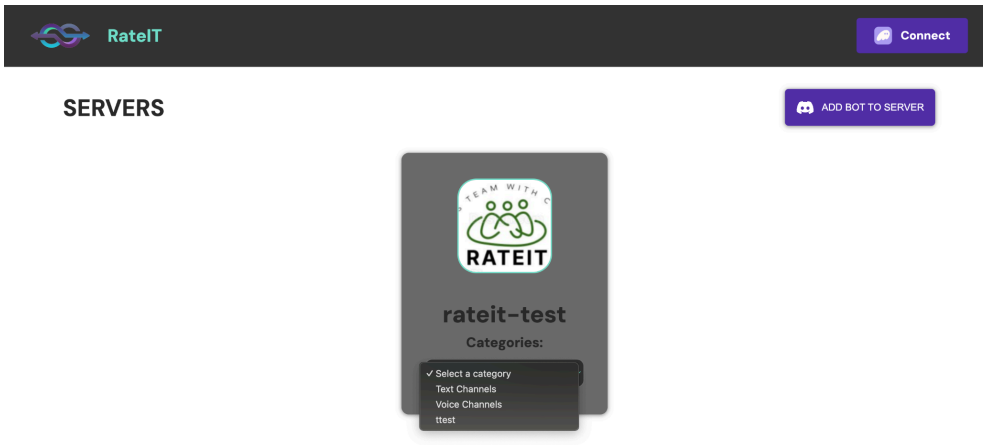

Рис. 3.4.2. *Вибір категорії на Discord сервері.*

Як тільки категорія обрана, нам відображаються, документи, які є в ній, також є опція додавання нового в форматі .md з депозитом в криптовалюті SOL. Саме на цьому моменті ми створюємо DAO “під капотом”.

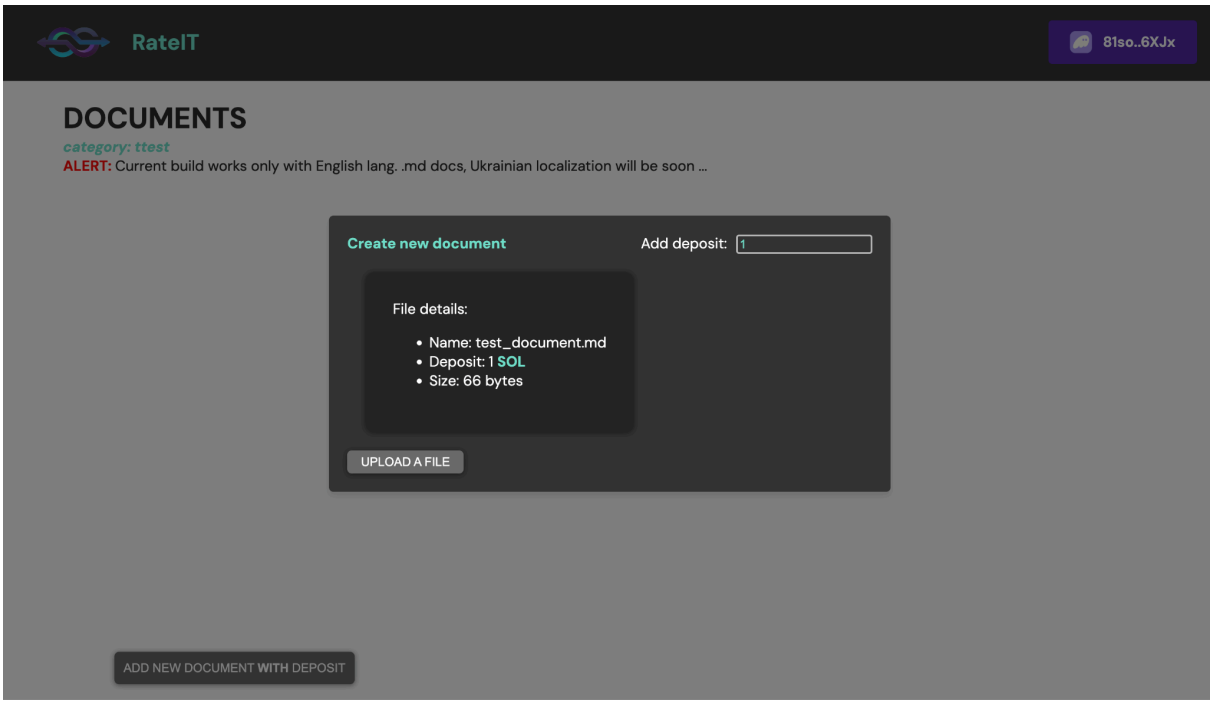

Рис. 3.4.3. *Створення нового документу з депозитом та DAO.*

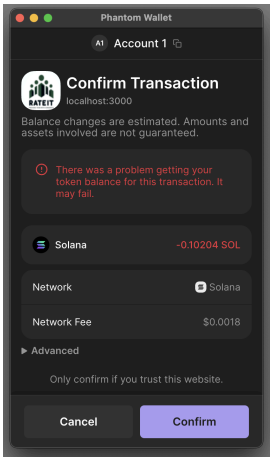

Рис. 3.4.4. *Транзакція, відображена в гаманці.*

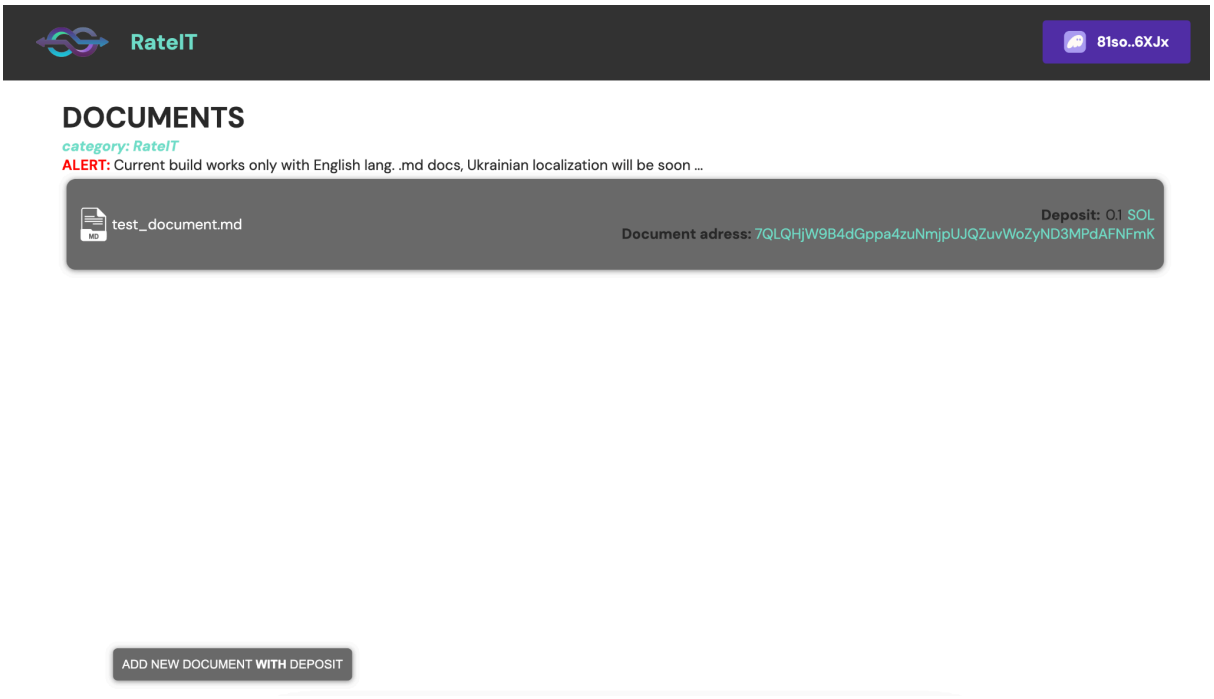

Рис. 3.4.5. *Відображення створеного документу.*

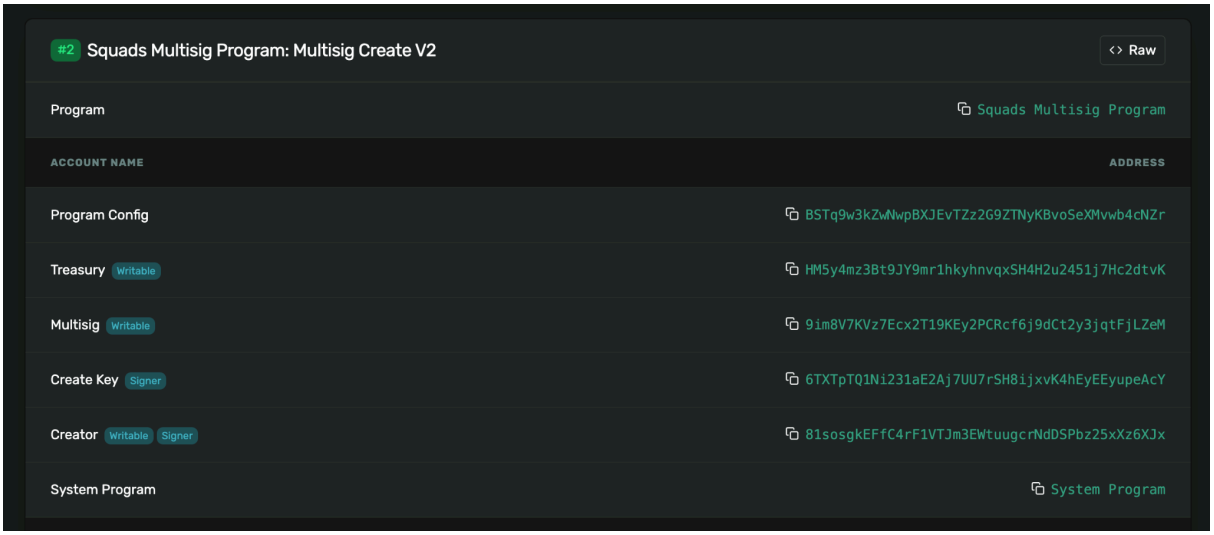

Рис. 3.4.5. *Інструкція транзакції створення DAO в блокчейні Solana.*

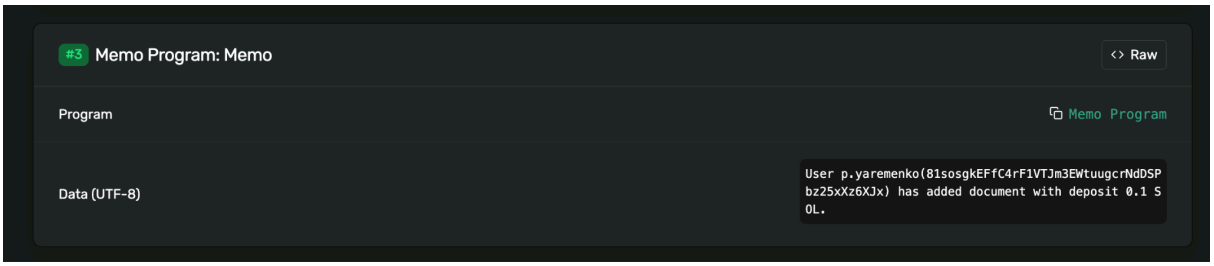

Рис. 3.4.6. *Підтвердження створення документу в блокчейні Solana.*

Після проведених маніпуляцій можемо починати розробку документу.

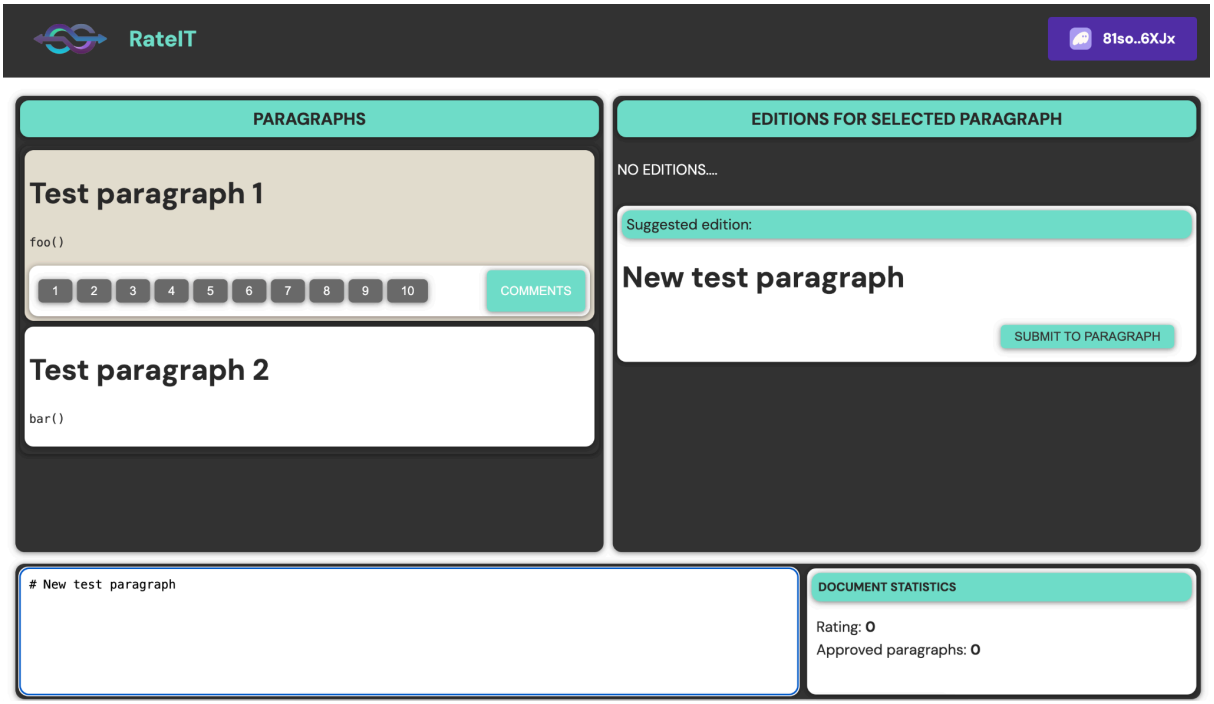

Рис. 3.4.7. *Сторінка роботи з документом.*

Рис. 3.4.7. *Меню для коментарів.*

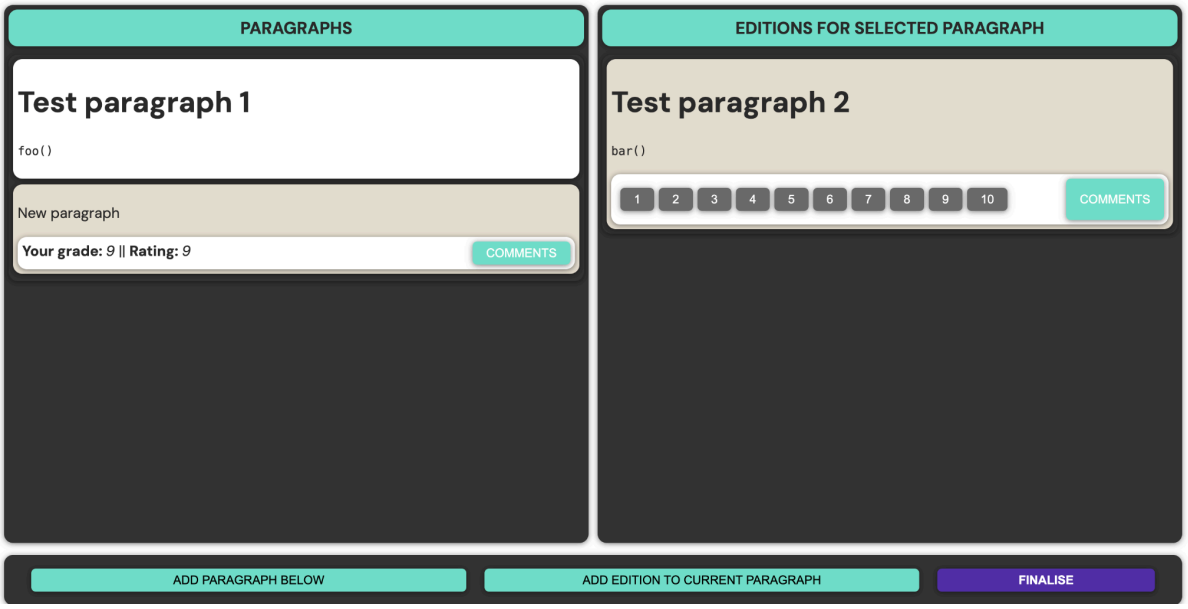

Рис. 3.4.8. *Оцінка редакцій.*

Рис. 3.4.9. *Відображення розподілу винагород.*

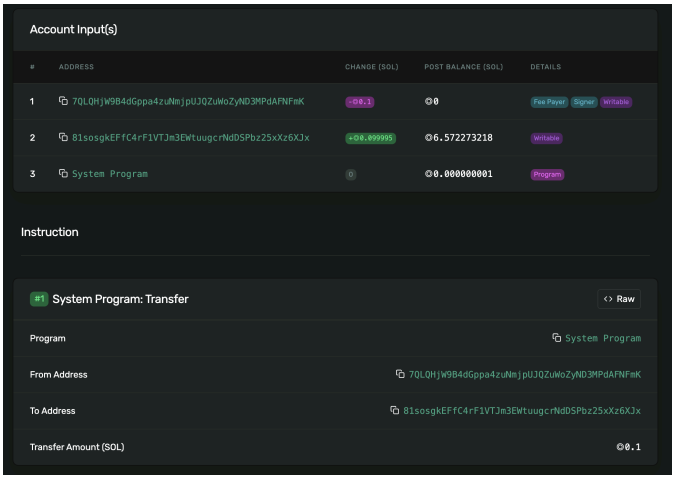

Рис. 3.4.9. *Відображення виконаної транзакції переказу винагороди на блокчейні Solana.*

Висновки

У процесі виконання кваліфікаційної роботи було досягнуто кілька важливих результатів, які підтвердили ефективність обраного підходу та значущість впроваджених рішень.

По-перше, розроблено програмний застосунок для колаборативної розробки документів у форматі ".md", який забезпечує зручність та ефективність спільної роботи. Застосунок дозволяє автоматично розбивати вхідні документи на окремі компоненти, які оцінюються та затверджуються всією командою. Це підвищує якість розробки та забезпечує прозорість процесу.

По-друге, імплементовано DAO та мультипідписні гаманці на базі блокчейну Solana, що дозволяє справедливо розподіляти винагороди між учасниками команди. Використання DAO сприяє децентралізації управління та забезпечує високий рівень безпеки та прозорості транзакцій. Також, розроблено та впроваджено власний алгоритм розподілу винагород, який враховує вклад кожного учасника у розробку документу. Це стимулює активну участь та підвищує мотивацію членів команди.

Науково-практичне значення дослідження полягало у демонстрації можливостей сучасних технологій для вирішення актуальних задач управління командною роботою та розподілу винагород. Отримані результати можуть бути використані для подальших досліджень та вдосконалення існуючих рішень у сфері колаборативної розробки.

Отже інноваційність роботи полягає у поєднанні сучасних технологій блокчейну та колаборативної розробки, що дозволяє створити ефективний інструмент для командної роботи. Було досліджено, що застосунок має високий потенціал для використання у різних галузях, де

колаборативна розробка документів є ключовим елементом. На даний момент це вже було перевірено та підтверджено, адже дана ідея отримала грант від Solana Foundation у розмір 9000 доларів США. Також даний проект брав участь у міжнародному хакатоні Solana Renaissance.

Список використаної літератури

1. Pieterse V., Kourie D. G., Sonnekus I. P. *Software engineering team diversity and performance. the 2006 annual research conference of the South African institute of computer scientists and information technologists*, Somerset West, South Africa, 9–11 October 2006. New York, New York, USA, 2006. URL: https://doi.org/10.1145/1216262.1216282.
2. Weyl E. G., Ohlhaver P., Buterin V. Decentralized Society: *Finding Web3's Soul. SSRN Electronic Journal.* 2022. URL: https://doi.org/10.2139/ssrn.4105763
3. Collaboration Software - Global | *Statista Market Forecast.* URL: https://www.statista.com/outlook/tmo/software/productivity-software/collaboration-software/worldwide.
4. Martin Nordio, H.-Christian Estler, Carlo A. Furia, and Bertrand Meyer: *Collaborative Software Development on the Web.* ETH Zurich, Switzerland, October 29, 2018. URL: https://arxiv.org/pdf/1105.0768
5. Scrum: An Effective Software Development Agile Tool / V. Hema et al. *IOP Conference Series: Materials Science and Engineering.* 2020. Vol. 981. P. 022060. URL: https://doi.org/10.1088/1757-899x/981/2/022060.
6. Hoffman M. R., Ibáñez L.-D., Simperl E. *Toward a Formal Scholarly Understanding of Blockchain-Mediated Decentralization: A Systematic Review and a Framework.* Frontiers in Blockchain. 2020. Vol. 3. URL: https://doi.org/10.3389/fbloc.2020.00035(date of access: 21.05.2024).
7. Новохацька Анастасія: *Система консенсусного погодження складних правок експертів, що працюють разом над законопроектами.* Національний університет «Києво-Могилянська академія», 2022

8. J.W. Bambacht and J.A. Pouwelse: Web3: *A Decentralized Societal Infrastructure for Identity, Trust, Money, and Data. Distributed Systems,* Delft University of Technology March 4, 2022. URL: https://arxiv.org/pdf/2203.00398
9. Longshak J. E. *The Emergence of Web3 and Metaverse Technologies.* Advances in Library and Information Science. 2022. P. 84–113. URL: https://doi.org/10.4018/978-1-6684-5964-5.ch007
10. Anatoly Yakovenko: Solana: A *New Architecture For A High Performance Blockchain*, 2020. URL: https://solana.com/solana-whitepaper.pdf
11. Amhaz R., Bobenrieth C., Marz M. *Impact of Decentralized Autonomous Organizations (DAO) on Society 5.0. 5th International Conference on Machine Learning, IOT and Blockchain.* 2024. URL: https://doi.org/10.5121/csit.2024.140301.
12. Squads Docs. *Welcome to Squads.* URL: https://docs.squads.so/main.
13. Zizi O. *Here's how much the crypto world loves Discord more than Telegram. The Business of Business - The leading business publication for the bold and ambitious | The Business of Business.* URL: https://www.businessofbusiness.com/articles/heres-how-much-the-crypto-world-loves-discord-more-than-telegram/.
14. Documentation. *NestJS - A progressive Node.js framework*. URL: https://docs.nestjs.com/.
15. Katja Hölttä, Eun Suk Suh, and Olivier de Weck: *TRADEOFF BETWEEN MODULARITY AND PERFORMANCE FOR ENGINEERED SYSTEMS AND PRODUCTS. INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN ICED 05,* MELBOURNE, AUGUST 15 – 18, 2005. URL: https://scholar.google.com.ua/scholar?q=Tradeoff+between+modularity+a

nd+performance+for+engineered+systems+and+products&hl=uk&as_sdt=0&as_vis=1&oi=scholart

16. Discord Docs for Bots and Developers. *Discord Developer Portal.* URL: https://discord.com/developers/docs/reference.
17. Introduction | *Solana Program Library Docs. Solana Docs.* URL: https://spl.solana.com/.
18. Worsley J. C., Drake J. D. *Practical PostgreSQL.* O'Reilly Media, Incorporated, 2002. URL: https://books.google.com.ua/books?hl=uk&lr=&id=fI11AgAAQBAJ&oi=fnd&pg=PR4&dq=postgresql&ots=a1UK58PWw6&sig=zZ3Zcvj3ZCgTLbIU0BQEEwCO4KM&redir_esc=y#v=onepage&q=postgresql&f=false
19. Documentation. *TypeORM.* URL: https://typeorm.io.
20. Harn L. *Group-oriented (t, n) threshold digital signature scheme and digital multisignature.* IEE Proceedings - Computers and Digital Techniques. 1994. Vol. 141, no. 5. P. 307. URL: https://doi.org/10.1049/ip-cdt:19941293.